Gowin タイミング制約

ユーザーガイド

SUG940-2.0.2J, 2025-12-31

著作権について(2025)
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、 、及び Gowin 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
2020/06/09	1.0J	初版。
2020/09/01	1.1J	● 温度グレード(オートモーティブ：-grade a)を追加。 ● 基本クロックと派生クロックの連動機能を追加。
2021/06/16	1.2J	● ワイルドカードの説明を追加。 ● 図面を更新。
2021/11/02	1.3J	● ソフトウェアのバージョンを更新。 ● 図面およびその説明を更新。 ● 「5 構文仕様」を更新。
2022/05/20	1.3.1J	一部の説明を更新。
2022/07/28	1.4J	● Create Clock に「-Add」オプションの説明を追加。 ● 遅延モデルの温度-電圧の説明を追加。 ● tUnc と tSu の説明を追加。 ● ワイルドカードの「階層間で一致」機能の説明を追加。
2022/12/16	1.4.1J	● セクション「I/O 遅延制約」のオプション Add delay の説明を更新。
2023/03/31	1.4.2J	● セクション「I/O 遅延制約」の set_input_delay と set_output_delay の説明を更新。 ● セクション「Total Negative Slack Summary」の説明を更新。
2023/04/20	1.4.3J	「図 3-8 New ウィンドウ」と「図 3-9 Open ウィンドウ」、およびその説明を追加。
2023/05/25	1.5J	● 例外制約の説明を更新。 ● ダミークロック DEFAULT_CLK の説明を更新。
2023/08/18	1.5.1J	● set_operation_conditions を set_operating_conditions を変更。 ● 最大周波数報告制約の説明を更新。
2023/11/30	1.6J	● クロックサイクルと周波数変換の説明を削除。 ● Process ウィンドウのスクリーンショットとその説明を削除。 ● set_output_delay の説明を更新。
2024/02/02	1.7J	derive_clocks 制約を追加。
2024/06/28	1.8J	create_clock 構文における-add オプションの使用例を追加。
2024/08/09	1.8.1J	「図 4-1 静的タイミング解析レポート」および「図 4-2 Timing Summaries」を更新。

日付	バージョン	説明
2024/10/25	1.8.2J	「Clock Summary」における Clock Name のデフォルト表示ルールの説明を改善。
2024/12/31	1.8.3J	● 「図 3-5 タイミング制約エディタの GUI」を更新。 ● 「5 構文仕様」の例を更新。
2025/04/30	1.9J	● 「5 構文仕様」の説明を更新。 ● 一部の図面およびその説明を更新。
2025/06/27	2.0J	● 差動 BUF におけるクロック生成ルールの説明を追加。 ● 最小パルス幅の計算式を追加。
2025/08/29	2.0.1J	STA レポートから tUnc 行の NODE 情報を削除。
2025/12/31	2.0.2J	● グループ制約の例 Example 5.16 と Example 5.17 を更新。 ● 派生クロック作成用の「Master Clock」を指定するルールを追加。 ● 「5.3 I/O 制約」に、「-add_delay」に対するソフトウェア解析ルールを追加。 ● 「図 3-27 Operating Conditions 制約の作成」を更新。 ● 動作条件の制約に、各シリーズ製品の Slow モデルと Fast モデルに対応する電圧と温度を追加。 ● create_clock、derive_clocks、および Configuration->Global->Frequency(MHz)の優先度を追加。

目次

目次 i
図一覧 iv
表一覧 vi
1 本マニュアルについて 1
1.1 主な内容 1
1.2 関連ドキュメント 1
1.3 用語、略語 1
1.4 テクニカル・サポートとフィードバック 2
2 概要 3
2.1 制約のサポート 3
2.2 レポートの出力 3
2.3 タイミングパス 4
3 タイミング制約エディタ 5
3.1 概要 5
3.2 タイミング制約エディタの起動 5
3.3 制約ファイルの新規作成、オープン、および追加 5
3.3.1 制約ファイルの新規作成 5
3.3.2 制約ファイルを開く 7
3.3.3 制約ファイルの追加 8
3.4 タイミング制約エディタの GUI 8
3.5 タイミング制約ウィンドウを開く 11
3.6 SDC ファイルの編集 12
3.7 タイミング制約の作成 12
3.7.1 クロック制約 13
3.7.2 I/O 遅延制約 22
3.7.3 例外制約 23
3.7.4 動作条件の制約 26
3.7.5 レポート制約 27

3.7.6 その他の制約 37
3.7.7 保存とエクスポート 38
3.8 タイミング制約の優先度 39
4 タイミングレポート 40
4.1 Timing Summaries 40
4.1.1 STA Tool Run Summary 41
4.1.2 Clock Summary 42
4.1.3 Max Frequency Summary 43
4.1.4 Total Negative Slack Summary 43
4.2 Timing Details 44
4.2.1 Path Slacks Table 44
4.2.2 Minimum Pulse Width Table 45
4.2.3 Timing Report By Analysis Type 46
4.2.4 Minimum Pulse Width Report 52
4.2.5 High Fanout Nets Report 53
4.2.6 Route Congestions Report 54
4.2.7 Timing Exceptions Report 54
4.2.8 Timing Constraints Report 57
5 構文仕様 59
5.1 オブジェクトのコレクション 59
5.2 クロック制約 59
5.2.1 create_clock 59
5.2.2 create_generated_clock 61
5.2.3 set_clock_latency 65
5.2.4 set_clock_uncertainty 67
5.2.5 set_clock_groups 70
5.3 I/O 制約 71
5.4 例外制約 75
5.4.1 set_false_path 75
5.4.2 set_max_delay / set_min_delay 76
5.4.3 set_multicycle_path 78
5.5 レポート制約 80
5.5.1 report_timing 80
5.5.2 report_high_fanout_nets 83
5.5.3 report_route_congestion 84
5.5.4 report_min_pulse_width 85
5.5.5 report_max_frequency 86

5.5.6 report_exceptions 87
5.6 動作条件の制約 89
5.7 その他の制約 90
5.7.1 derive_clocks 90

図一覧

図 3-1 制約ファイル新規作成ダイアログボックスを開く 6
図 3-2 タイミング制約ファイルの新規作成 6
図 3-3 タイミング制約ファイルを開く 7
図 3-4 タイミング制約ファイルの追加 8
図 3-5 タイミング制約エディタの GUI 9
図 3-6 Netlist Tree ウィンドウ 9
図 3-7 制約編集ウィンドウ 10
図 3-8 New ウィンドウ 10
図 3-9 Open ウィンドウ 11
図 3-10 メニューバーからタイミング制約ウィンドウを開く 11
図 3-11 右クリックしてタイミング制約ウィンドウを開く 12
図 3-12 SDC ファイルの編集 12
図 3-13 基本クロックの作成 13
図 3-14 オブジェクトの選択 14
図 3-15 クロックの追加 15
図 3-16 クロックリスト 15
図 3-17 クロックリストの右クリック項目 15
図 3-18 派生クロック制約の作成 17
図 3-19Create Generated Clock を選択 18
図 3-20 クロック遅延時間の設定 19
図 3-21 ばらつきの設定 20
図 3-22 クロックグループの設定 21
図 3-23 I/O Delay 制約の作成 23
図 3-24 False Path 制約の作成 24
図 3-25 Max/Min Delay 制約の作成 25
図 3-26 Multicycle Path 制約の作成 26
図 3-27 Operating Conditions 制約の作成 27
図 3-28 Report Timing の新規作成 28
図 3-29 Report Timing ダイアログボックス 29
図 3-30 Report High Fanout Nets の作成 30

図 3-31 Report High Fanout Nets ダイアログボックス 31
図 3-32 Report Route Congestion の作成 32
図 3-33 Report Route Congestion ダイアログボックス 32
図 3-34 Report Min Pulse Width の作成 33
図 3-35 Report Min Pulse Width ダイアログボックス 34
図 3-36 Report Max Frequency の作成 35
図 3-37 Report Max Frequency ダイアログボックス 35
図 3-38 Report Exception の作成 36
図 3-39 Report Exception ダイアログボックス 36
図 3-40 Derive Clocks の作成 37
図 3-41 Create Derive Clocks を選択 38
図 3-42 Derive Clocks リスト 38
図 4-1 静的タイミング解析レポート 40
図 4-2 Timing Summaries 41
図 4-3 Path & Endpoints 42
図 4-4 Path Slacks Table 45
図 4-5 Minimum Pulse Width Table 46
図 4-6 Path Summary 47
図 4-7 Data Arrival Path 48
図 4-8 Data Required Path 48
図 4-9 Path Statistics 49
図 4-10 Hold Analysis Report 50
図 4-11 Recovery Analysis Report 51
図 4-12 Removal Analysis Report 52
図 4-13 Minimum Pulse Width Report 53
図 4-14 High Fanout Nets Report 53
図 4-15 Route Congestions Report 54
図 4-16 テストケース 54
図 4-17 Timing Exceptions 制約 55
図 4-18 Timing Exceptions Report 56
図 4-19 report_exception 文 57
図 4-20 report_exception レポート 57
図 4-21 Timing Constraints Report 58

表一覧

表 1-1 用語、略語 1
表 5-1 コレクションの説明 59
表 5-2 create_clock のパラメータ 60
表 5-3 create_generated_clock のパラメータ 62
表 5-4 set_clock_latency のパラメータ 65
表 5-5 set_clock_uncertainty のパラメータ 67
表 5-6 set_clock_groups のパラメータ 70
表 5-7 解析結果 70
表 5-8 解析結果 71
表 5-9 I/O 制約のパラメータ 72
表 5-10 set_false_path のパラメータ 75
表 5-11 set_max_delay / set_min_delay のパラメータ 77
表 5-12 set_multicycle_path のパラメータ 78
表 5-13 report_timing のパラメータ 80
表 5-14 report_high_fanout_nets のパラメータ 83
表 5-15 report_route_congestion のパラメータ 84
表 5-16 report_min_pulse_width のパラメータ 85
表 5-17 report_max_frequency のパラメータ 87
表 5-18 report_exceptions のパラメータ 88
表 5-19 動作条件制約パラメータ 89
表 5-20 derive_clocks のパラメータ 91

1 本マニュアルについて

1.1 主な内容

このマニュアルは主に、Gowin のタイミング制約エディタ、タイミングレポート、および構文仕様について解説します。

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 クイックスタートガイド(SUG918)。

1.3 用語、略語

表 1-1 に、本マニュアルで使用される用語、略語、及びその意味を示します。

表 1-1 用語、略語

用語、略語	正式名称	意味
FF	Flip-Flop	フリップフロップ
GUI	Graphical User Interface	グラフィカル・ユーザー・インターフェース
I/O	Input and output	入出力ポート
LSR	Clear and set and reset and preset	クリア、セット、リセット
NET	Net	ネット
OSC	Oscillator	発振器
PIN	Pin	インスタンス化されたモジュールの入出力ピン
PnR	Place and Route	配置と配線

用語、略語	正式名称	意味
PORT	Port	トップ・モジュールの入出力ポート
SDC	Synopsys Design Constraint	Synopsys 設計制約
STA	Static Timing Analysis	静的タイミング解析

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.cn

E-mail：support@gowinsemi.com

2 概要

Gowin 静的タイミング解析(STA)は、回路ネットリストのタイミングパスに全面的な解析を行い、回路のタイミングパスの遅延を計算することにより、それが要件を満たしているかどうかを判断するものです。Gowin ソフトウェアにより計算と解析が自動的に実行されるので、設計者は制約スティミュラスを提供するだけです。詳細なタイミングレポートが提供され、検証時間やカバレッジにも大きな利点があります。

2.1 制約のサポート

- クロック制約をサポート。
- I/O 遅延制約をサポート。
- 例外制約をサポート。
- レポート制約をサポート。
- 動作条件制約をサポート。
- グローバル・クロック制約をサポート。
- タイミング制約エディタ GUI による制約の編集をサポート。
- タイミング制約構文は、SDC 仕様を参照。

2.2 レポートの出力

- タイミング遅延モデルの出力をサポート。
- 最大クロック周波数の出力をサポート。
- セットアップ(setup)とホールド(hold)パス解析の出力をサポート。
- リカバリ（recovery）とリムーバル（removal）パス解析の出力をサポート。
- 最小パルス幅解析の出力をサポート。

- 高ファンアウト net の出力をサポート。
- 配線の密集レベルの出力をサポート。
- 例外制約レポートの出力をサポート。
- HTML および TXT ファイル形式の出力をサポート。

2.3 タイミングパス

静的タイミング解析は 4 種類のタイミングパスを解析します。これらのパスは、始点と終点によって分類されています。

- 入力ポートから内部シーケンシャル・エレメントまでのパス。
- シーケンシャル・エレメントからシーケンシャル・エレメントまでのパス。
- 内部シーケンシャル・エレメントから出力ポートまでのパス。
- 入力ポートから出力ポートまでのパス。

3 タイミング制約エディタ

3.1 概要

タイミング制約エディタ(Timing Constraints Editor)を使用してタイミング制約を追加、削除、変更、および確認できます。

3.2 タイミング制約エディタの起動

タイミング制約エディタは、単独で使用するか、または合成後に使用することができます。

単独で使用する場合は、「Tools> Timing Constraints Editor」をクリックして起動します。合成後に使用する場合は、Gowin ソフトウェアの Process ウィンドウで「Synthesize」を正常に実行した後、「Process > Timing Constraints Editor」をダブルクリックして起動します。ネットリスト・ファイルとタイミング制約ファイルは、タイミング制約エディタに自動的にロードされます。プロジェクトにタイミング制約ファイルがない場合は、自動的に作成されます。

3.3 制約ファイルの新規作成、オープン、および追加

3.3.1 制約ファイルの新規作成

以下は、制約ファイルの新規作成手順です。

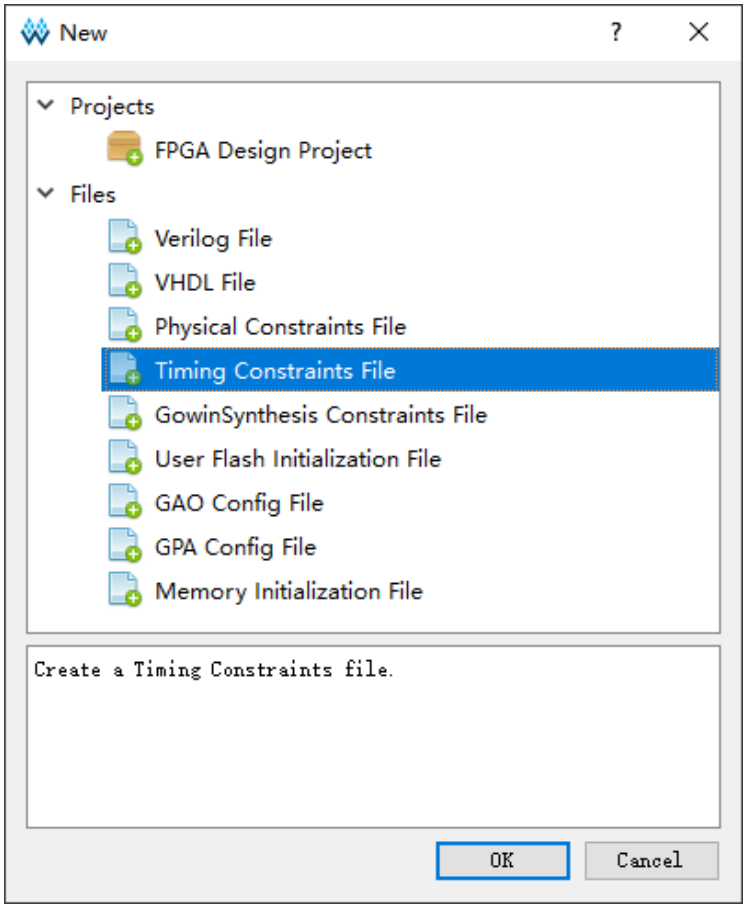
1. 「File>New」メニューをクリックすると、ファイル新規作成のダイアログボックスが開きます。
2. 「Timing Constraints File」オプションを選択します(図 3-1)。

または、以下の方法で制約ファイル新規作成ダイアログボックスを開きます。

- ツールバーの「New」アイコンをクリックします。

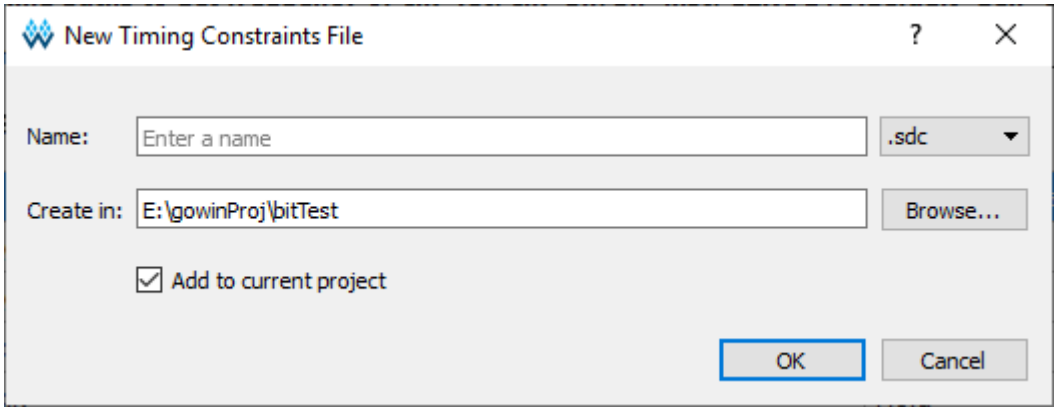
- ショートカットキーCtrl + N を使用します。

図 3-1 制約ファイル新規作成ダイアログボックスを開く



3. 「OK」をクリックすると、タイミング制約ファイル新規作成ダイアログボックスがポップアップします(図 3-2)。

図 3-2 タイミング制約ファイルの新規作成



4. ファイル名を入力して作成ディレクトリを選択し、「OK」をクリックすると、タイミング制約ファイルは作成されてプロジェクトに自動的にロードされます。

- Name：新規作成されたタイミング制約ファイルの名前。.sdc ファイルがサポートされます。
- Create in：「Browse」ボタンをクリックして新しい制約ファイルの保存場所を選択します。デフォルトのパスはプロジェクトディレクトリの src フォルダです。
- Add to current project：このオプションを選択すると、制約ファイルはプロジェクトに自動追加されます。デフォルトではチェックされています。

3.3.2 制約ファイルを開く

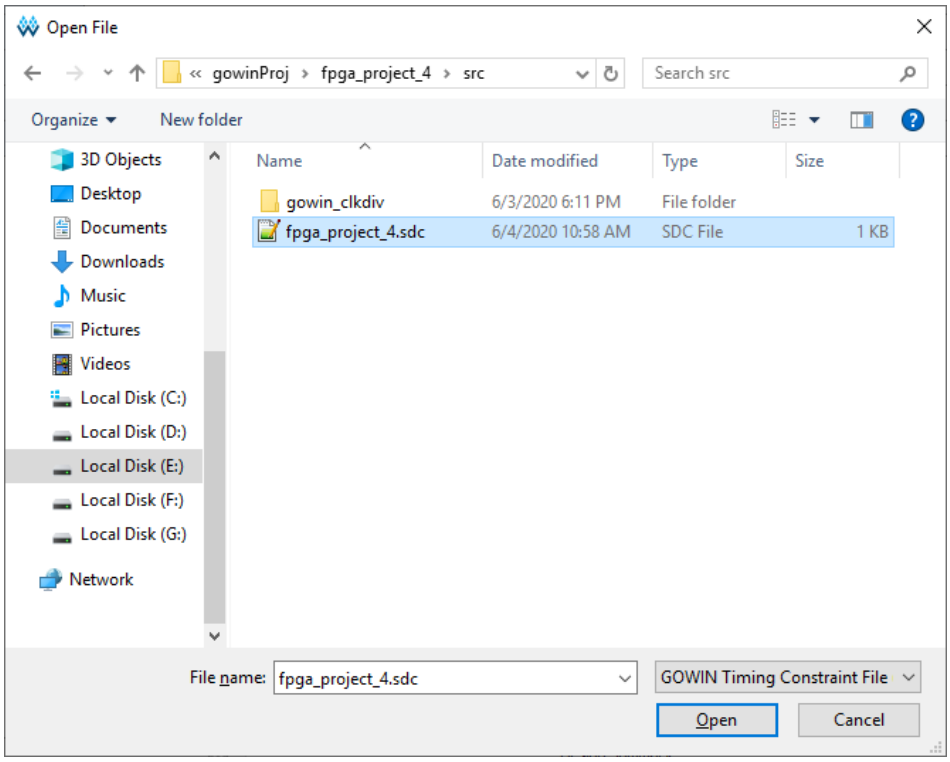
以下は、制約ファイルを開く手順です。

1. IDE で「File>Open」をクリックします。
2. 「Open File」ダイアログボックスを開きます(図 3-3)。

または、以下の方法を使用します。

- ツールバーの「Open」アイコンをクリックします。
- ショートカットキーCtrl + O を使用します。

図 3-3 タイミング制約ファイルを開く



3. タイミング制約ファイルが所在するディレクトリでファイルを選択して「Open」をクリックします。sdc ファイルがサポートされます。

3.3.3 制約ファイルの追加

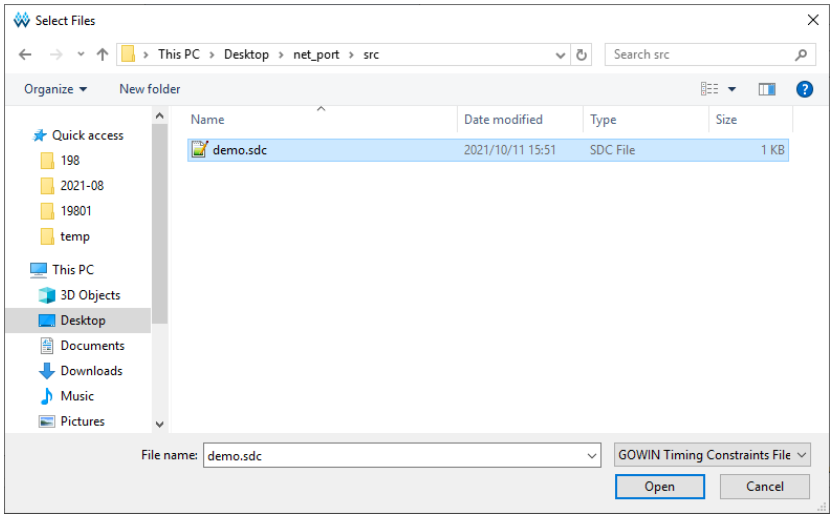
以下は、制約ファイルを追加する手順です。

1. IDE の Design ウィンドウで右クリックして、「Add Files」を選択します。
2. ポップアップする「Select Files」ダイアログボックスから「.sdc」ファイルタイプを選択します(図 3-4)。
3. 制約ファイルを選択したら、「Open」をクリックしてプロジェクトに追加します。

注記：

複数のファイルを追加した場合、デフォルトで最後に追加されたファイルが有効です。

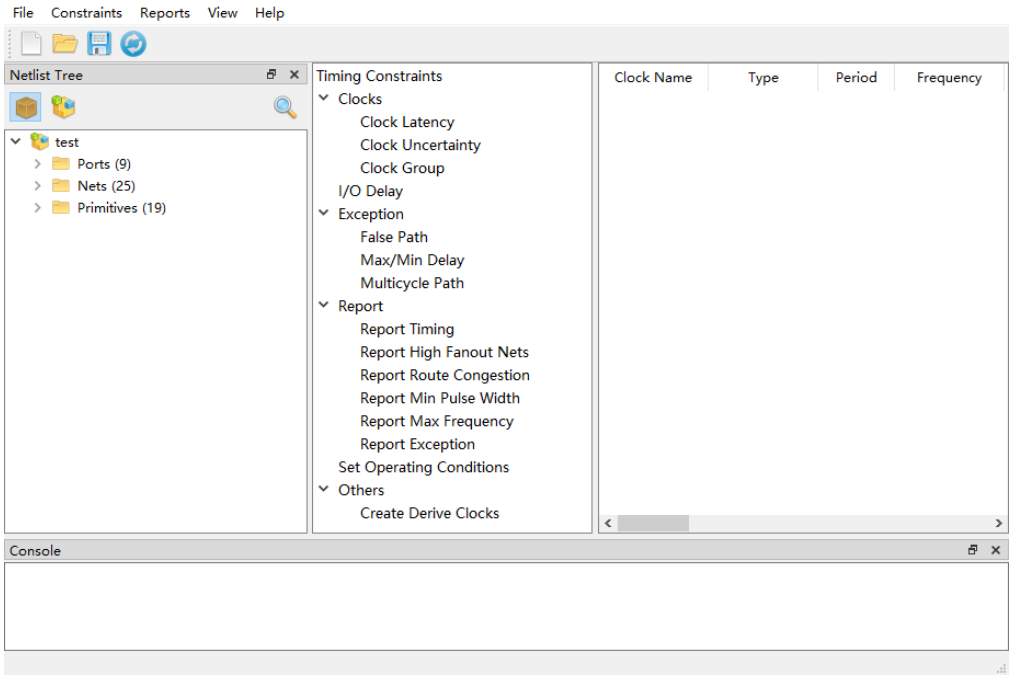
図 3-4 タイミング制約ファイルの追加



3.4 タイミング制約エディタの GUI

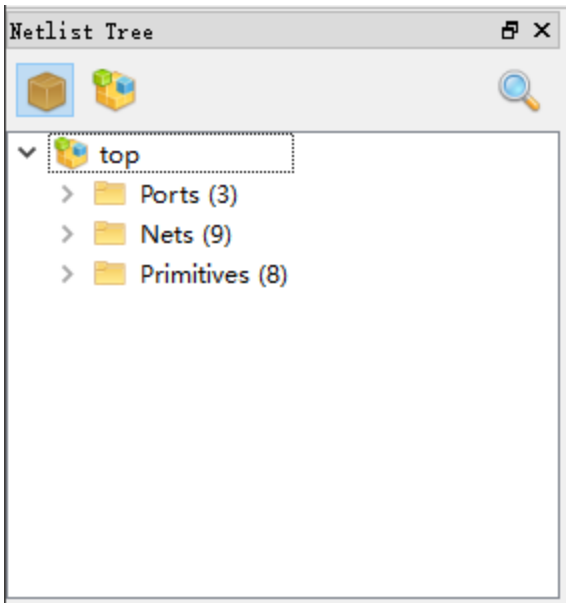
図 3-5 は、制約ファイルを開いた後のタイミング制約エディタの GUI です。

図 3-5 タイミング制約エディタの GUI



メインウィンドウの左上隅は、Netlist Tree ウィンドウです(図 3-6)。

図 3-6 Netlist Tree ウィンドウ



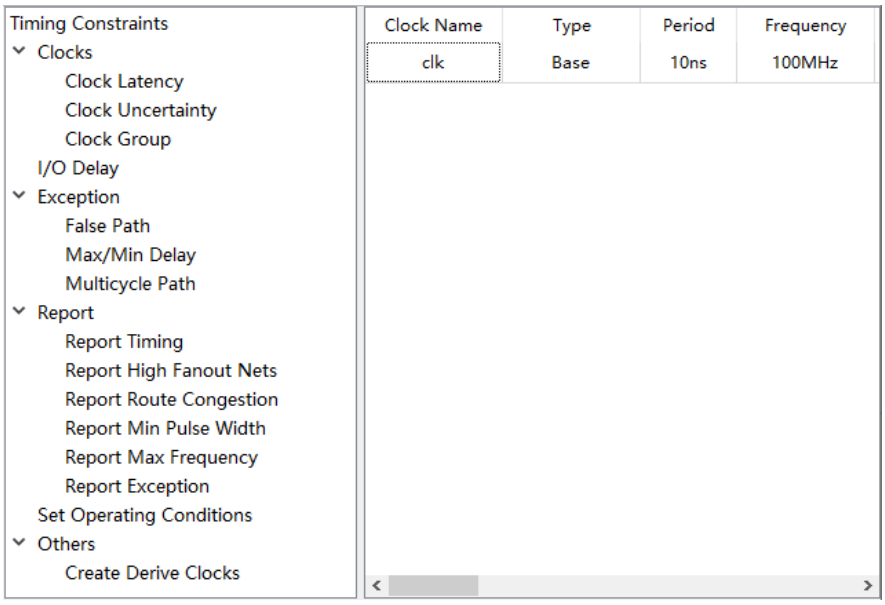
Netlist Tree ウィンドウには、現在のネットリスト・ファイルの Top Module、I/O Ports、Nets、Primitives が含まれています。

- 「 」: flatten リストを確認します。

- 「 」: hierarchy リストを確認します。

メインウィンドウの中央及び右側のエリアは、制約編集エリアです(図 3-7)。そのうち左側のリストはタイミング制約タイプの一覧で、右側は表の編集エリアです。タイプの一覧で任意の制約タイプをクリックすると、表の編集エリアで設定済みの制約編集リストが表示されます。

図 3-7 制約編集ウィンドウ



メインウィンドウの上部のツールバーには、「New 」、「Open 」、「Save 」、「Reload 」というボタンがあります。New ウィンドウでは、ネットリスト・ファイル「Input Netlist File」とデバイス情報「Device」を選択することができます。

図 3-8 New ウィンドウ

Open ウィンドウでは、ネットリスト・ファイル「Input Netlist File」、制約ファイル「Constraint File」、およびデバイス情報「Device」を選択することができます。

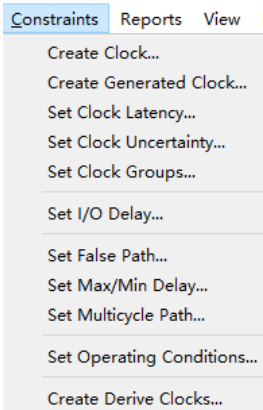
図 3-9 Open ウィンドウ

3.5 タイミング制約ウィンドウを開く

タイミング制約ウィンドウを開く方法は 2 つあります。

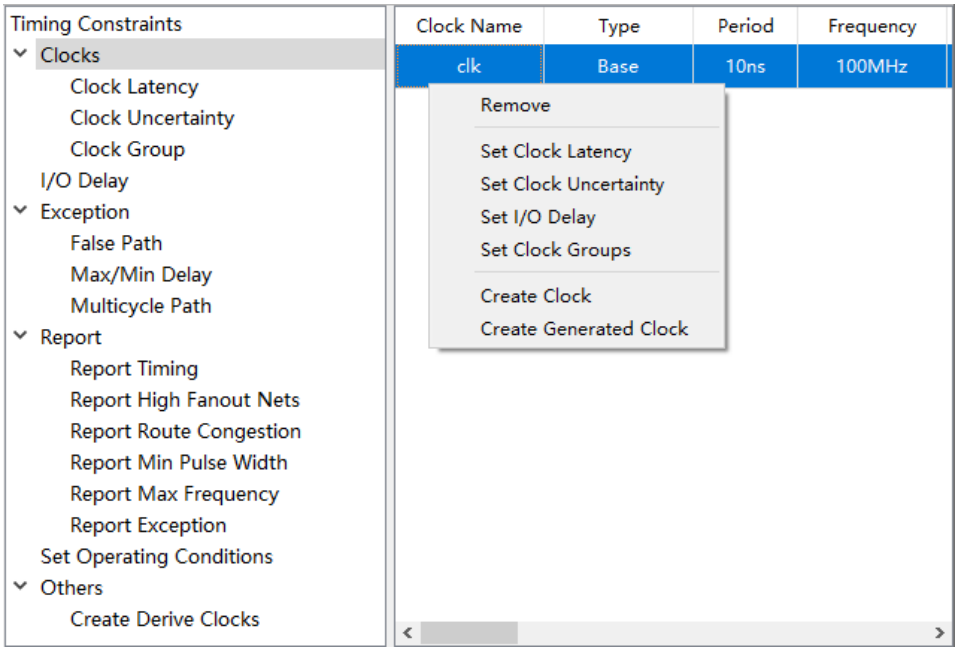
1. メニューバーで「Constraints」をクリックし、ドロップダウン・リストからタイミング制約コマンドを選択して対応するタイミング制約ウィンドウを開きます(図 3-10)。

図 3-10 メニューバーからタイミング制約ウィンドウを開く



2. タイミング制約エディタの右側にある表で右クリックしてメニューからさまざまなタイミング制約コマンドを選択して対応するタイミング制約ウィンドウを開きます(図 3-11)。

図 3-11 右クリックしてタイミング制約ウィンドウを開く



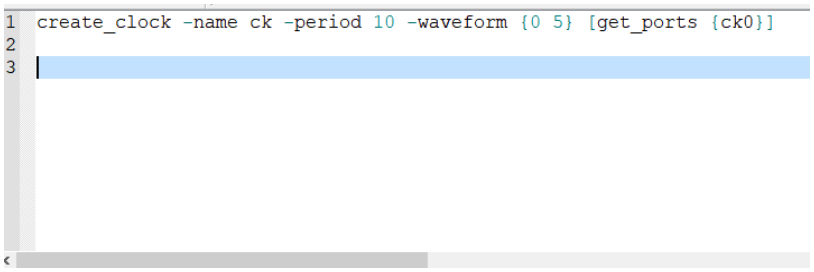
3.6 SDC ファイルの編集

Gowin ソフトウェアでは、プロジェクトの SDC ファイルを読み出し、テキストエディタで制約を簡単に手動変更することができます(図 3-12)。

SDC ファイルの解析はワイルドカードをサポートします。現在「*」と「?」の 2 つのワイルドカードがサポートされています。「*」は 0 文字以上の一致を実現し、「?」は 1 文字の一致を実現します。

SDC ファイルは、単一行コメントと複数行コメントをサポートします。単一行コメントの場合は「//」または「#」を使用し、複数行コメントの場合は「/* */」を使用します。

図 3-12 SDC ファイルの編集



3.7 タイミング制約の作成

このセクションでは、タイミング制約エディタを使用してタイミング制約を作成する方法を紹介します。作成された制約は Console ウィンドウに出力され、「Save」をクリックすると SDC ファイルに保存されま

す。

3.7.1 クロック制約

Create Clock

クロックを作成します。

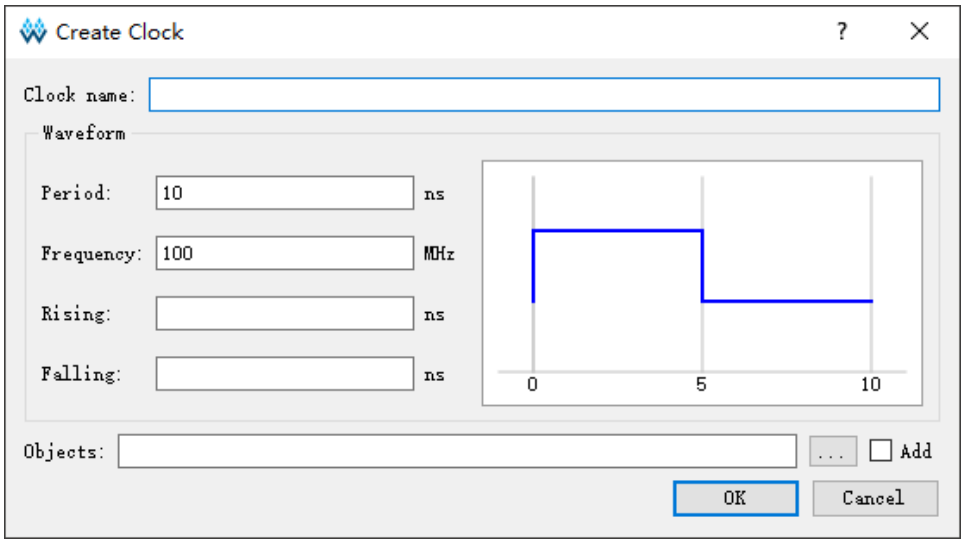
クロックの名前、周期、周波数、立ち上がりエッジ、立ち下がりエッジ、およびクロックのオブジェクトなどのパラメータを指定できます。Gowin ソフトウェアは、複数のクロックドメインの確立をサポートし、クロスクロックドメイン解析をサポートます。

Clock 制約を追加するには、2 つの方法があります。

1. 「Constraints」メニューで Clock 制約を追加します。

a). 「Constraints > Create Clock…」を選択すると「Create Clock」ダイアログボックスがポップアップします(図 3-13)。

図 3-13 基本クロックの作成



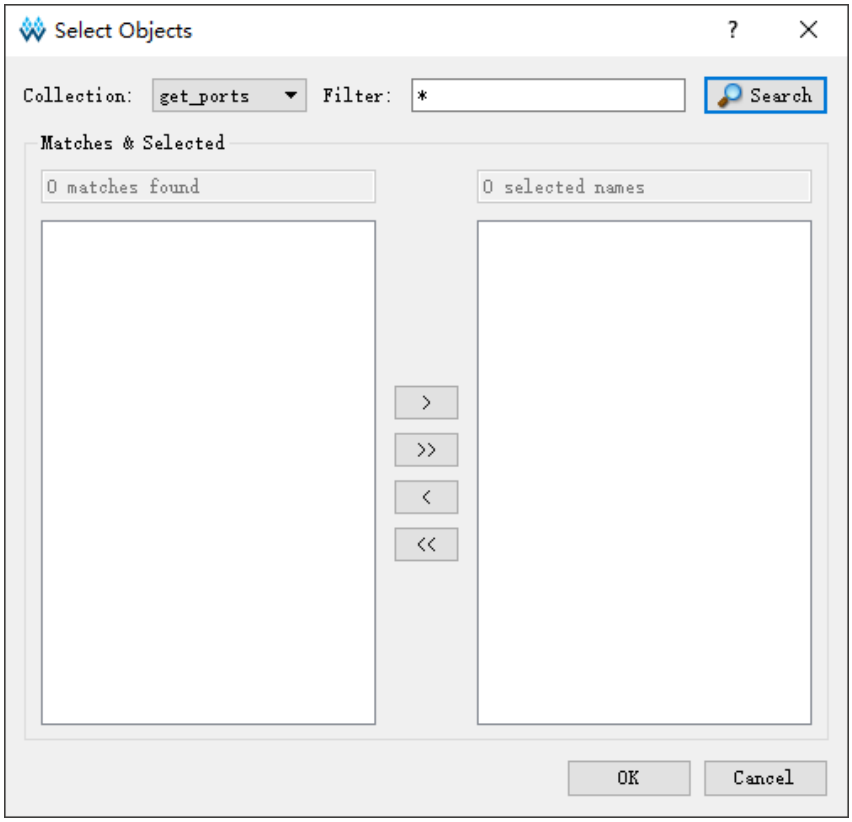
b). Clock の情報を入力します。

- Clock Name：クロック名。文字またはアンダースコアで始まることをサポートします。
- Period：周期、デフォルトは 10。0 より大きい浮動小数点数型、小数点以下 3 桁、単位は ns。
- Frequency：周波数、デフォルトは 100。0 より大きい浮動小数点数型、小数点以下 3 桁、単位は MHz。
- Rising：立ち上がりエッジ時点。0 以上の浮動小数点数型、小数点以下 3 桁、単位は ns。

- Falling：立ち下がりエッジ時点。0 より大きい浮動小数点数型、小数点以下 3 桁、単位は ns。
- Objects：オブジェクトを指定します。「...」をクリックして選択します。
- Add：同じソースに複数のクロックを追加する場合は、チェックする必要があります。

c). Objects 右側の「...」ボタンをクリックすると、「Select Objects」ダイアログボックスがポップアップします(図 3-14)。

図 3-14 オブジェクトの選択



d). 図 3-14 では、「Collection」は検索されるコレクションタイプを指定します。「Filter」はフィルタです。「Search」をクリックすると、すべての該当する結果トは左側に表示されます。「>」ボタンをクリックすると、選択されたアイテムは左側のリストから右側のリストに追加されます。「>>」をクリックすると、すべてのアイテムは右側のリストに追加されます。「<」をクリックすると、右側の選択されたアイテムが削除されます。「<<」をクリックすると、右側のすべてのアイテムが削除されます。

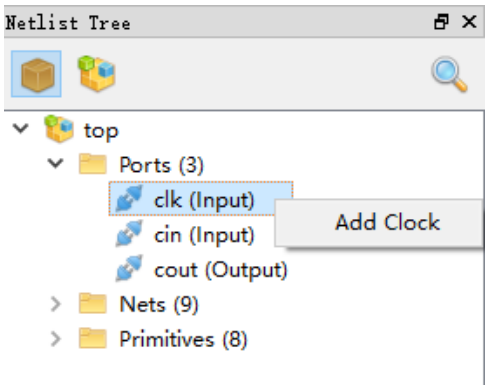
e). 「OK」をクリックして Objects の追加を完了します。

2. Netlist Tree で Clock 制約を追加します。

a). Netlist Tree で Port または Net を選択します。

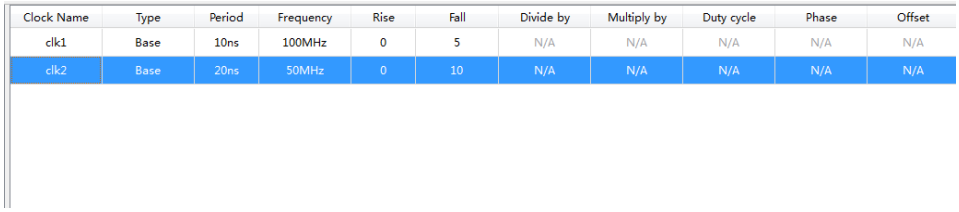
b). 右クリックして「Add Clock」を選択し、クロックを追加します(図 3-15)。

図 3-15 クロックの追加



作成されたクロックは図 3-16 に示すとおりです。

図 3-16 クロックリスト

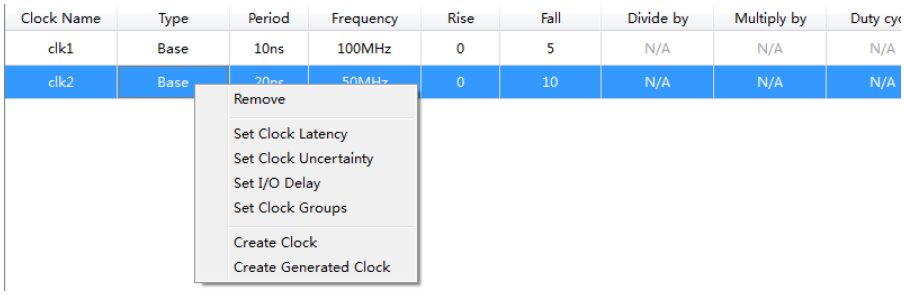


Clock Name	Type	Period	Frequency	Rise	Fall	Divide by	Multiply by	Duty cycle	Phase	Offset
clk1	Base	10ns	100MHz	0	5	N/A	N/A	N/A	N/A	N/A
clk2	Base	20ns	50MHz	0	10	N/A	N/A	N/A	N/A	N/A

このリストでは、以下の操作を行うことができます。

- Clock の編集。「Clocks」リストの対応する制約をダブルクリックすると Clock の編集ダイアログボックスが開くので、ダイアログで Clock 情報を編集できます。
- Clock の削除。リストでその Clock を右クリックして「Remove」を選択します。
- Clock を右クリックしてその Clock の Clock Latency、Clock Uncertainty、I/O Delay 情報を素早く設定できます(図 3-17)。

図 3-17 クロックリストの右クリック項目



注記：

- 制約が PLL の構成と矛盾する場合、Create Clock によって作成された制約が優先されます。配置配線中に警告メッセージが表示されます。

- Create Clock は、ダミークロックの作成をサポートしていません。
- 差動 BUF の N 側(IOB など)にクロックを作成することはサポートされていません。
- Gowin は、異なるクロック・デューティサイクルに基づくタイミング解析をサポートしています。

Create Generated Clock

派生クロックを作成します。

この制約により、マスタークロックに基づいて分周と逓倍、位相シフト、デューティサイクル調整などを実行して派生クロックを作成することができます。作成された派生クロックは自動的にマスタークロックと連動します。マスタークロックの属性が変更されると、派生クロックもそれに応じて自動的に変更されます。

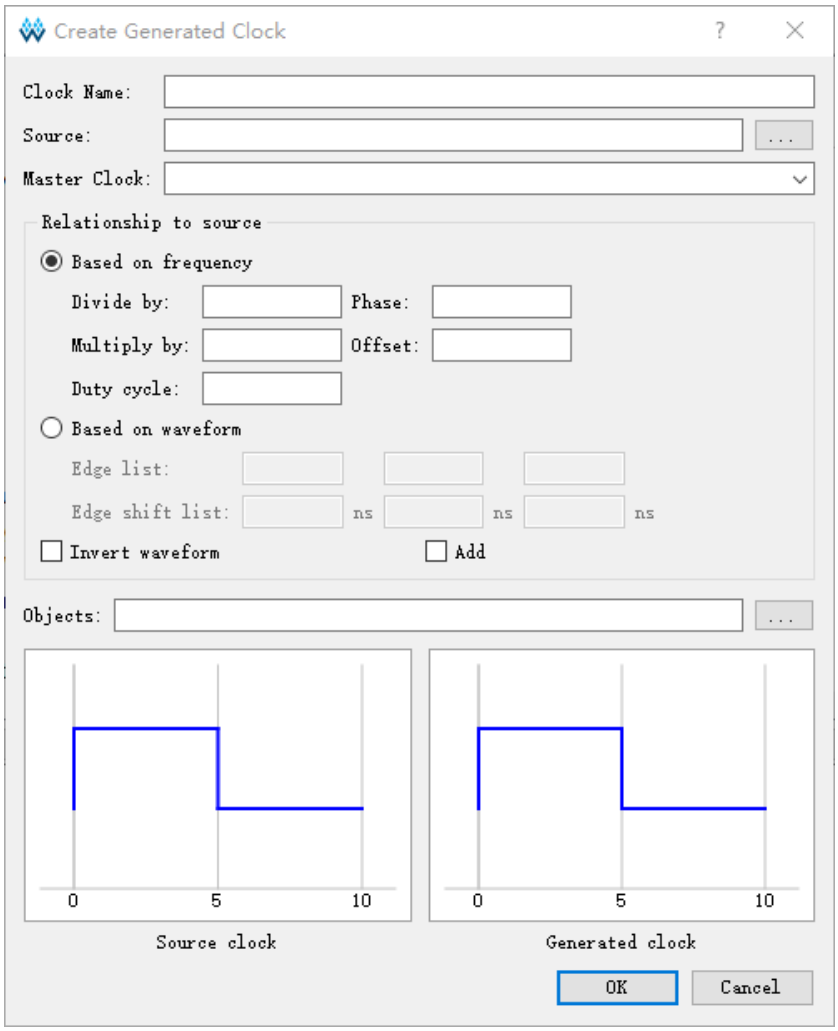
以下の 2 つの方法で派生クロックを新規作成できます。

1. 「Constraints」メニューで作成します。

a). 「Constraints」メニューで「Create Generated Clock」を選択すると「Create Generated Clock」ダイアログボックスがポップアップします(図 3-18)。

- Clock Name：クロック名。文字またはアンダースコアで始まることをサポートします。
- Source：派生クロックのソース。右側の「...」をクリックして選択します。
- Master Clock：マスタークロック。右側の「⌄」をクリックして選択します。

図 3-18 派生クロック制約の作成



b). Source の右側にある「...」をクリックして、クロックのソースを選択します。「Source」に関連付けられたクロックは、「Master Clock」リストに自動的に追加されます。次に「Master Clock」を選択します。「Master Clock」に複数のクロックがある場合、1 つのみを選択できます。「Master Clock」として指定されるすべてのクロックは同じコレクションに属している必要があります。

c). 「Relationship to source」では、「Base on frequency」の場合は、現在作成されている派生クロックに対して周波数分割、逓倍、オフセット、デューティサイクル、および位相の調整などを実行できます。「Base on waveform」の場合は、Edge list および Edge shift list を併用することにより、派生クロックのエッジ調整を実現できます。

- Divide by：分周値。正の整数。
- Phase：位相。浮動小数点数型、小数点以下 3 桁。負の数の場合は左にシフトされ、正の数の場合は右にシフトされます。

- Multiply by：逓倍値。正の整数。
- Offset：オフセット。浮動小数点数型、小数点以下 3 桁。負の数の場合は左にシフトされ、正の数の場合は右にシフトされます。
- Duty cycle：デューティサイクル。浮動小数点数型、小数点以下 3 桁。
- Edge list：順番に増加する正の整数。
- Edge shift list：浮動小数点数型、小数点以下 3 桁。

d). 「Invert waveform」はクロックの反転を実現します。「Add」はさらなる追加を実現できます。これは STA の際にも有効です。

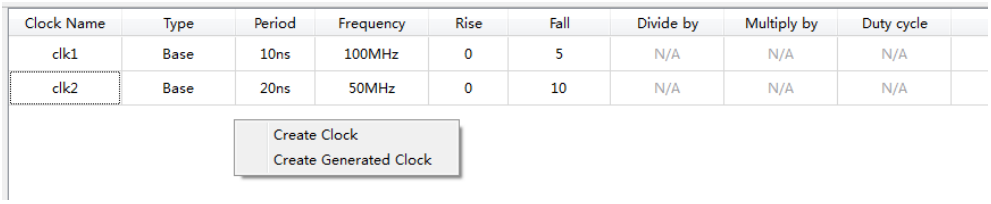
e). Objects の右側にある「...」ボタンをクリックし、「Select Objects」ダイアログボックスがポップアップします。オブジェクトを選択します。

注記：

- 制約が PLL の構成と矛盾する場合、Create Generated Clock によって作成された制約が優先され、配置配線中にも警告メッセージが表示されます。

2. Clocks リストで Generated Clock を作成します。Clocks リストの空白部分を右クリックし、「Create Generated Clock」を選択して Generated Clock を作成します(図 3-19)。

図 3-19Create Generated Clock を選択



追加後、テーブルの編集エリアに新規作成された制約が追加されます。

このリストでは、以下の操作を行うことができます。

- Generated Clock 制約の編集。「Clocks」リストの対応する制約をダブルクリックすると Generated Clock の編集ダイアログボックスが開くので、ダイアログで Generated Clock 情報を編集できます。
- Generated Clock の削除。表編集エリアでこの C l ock を選択し、右クリックで「Remove」を選択します。

Set Clock Latency

クロック遅延時間を設定します。

クロック遅延時間には、ネットワーク(NETWORK)遅延時間とソース

(SOURCE)遅延時間の 2 種類があります。

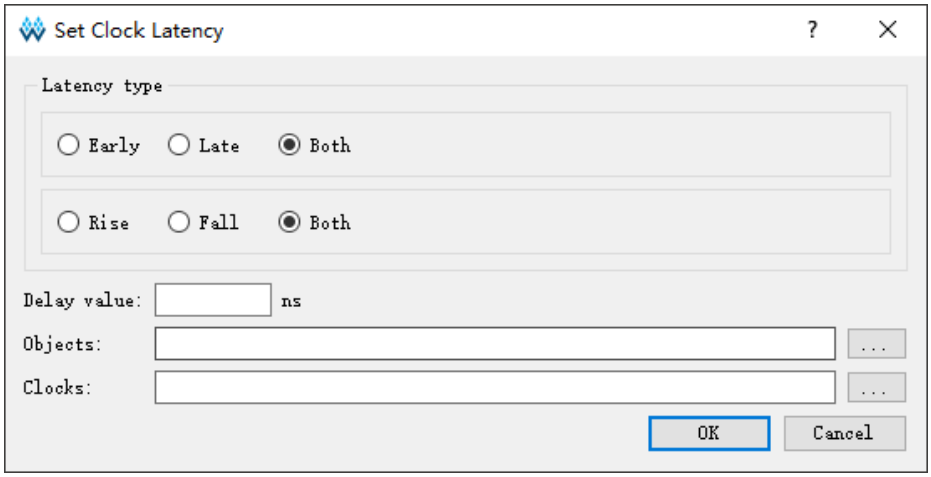
- ネットワーク遅延時間は、クロックとシーケンシャル・エレメント間の遅延時間です。
- ソース遅延時間は、クロックとソース間の遅延時間です。

クロックのネットワーク遅延時間は Gowin ソフトウェアにより自動的に計算されるため、ユーザーはソース遅延時間を設定するのみです。

以下の 2 つの方法で Clock Latency 制約を新規作成できます。

1. 「Constraints」メニューで Clock Latency 制約を新規作成します。
   「Constraints」メニューで「Set Clock Latency」を選択すると、「Set Clock Latency」ダイアログボックスがポップアップします(図 3-20)。Latency 情報を入力し、「OK」をクリックして制約を保存します。
   - Early、Late：それぞれ最小遅延と最大遅延を示します。Both は両者有効を示します。
   - Rise と Fall：それぞれ立ち上がりエッジ有効と立ち下がり有効を示します。Both は両者有効を示します。
   - Delay value：クロックの遅延時間値。浮動小数点数型、小数点以下 3 桁。単位は ns。
   - Objects：右側の「...」をクリックして選択します。オブジェクトを指定します。
   - Clocks：右側の「...」をクリックして選択します。対象クロックを指定します。

図 3-20 クロック遅延時間の設定



2. Clocks リストから Clock Latency 制約を新規作成します。

   Clocks リストで Clock を右クリックして Set Clock Latency を選択し、この Clock の Latency 情報を設定します。Objects は自動的にこのク

ロックとして指定されます。

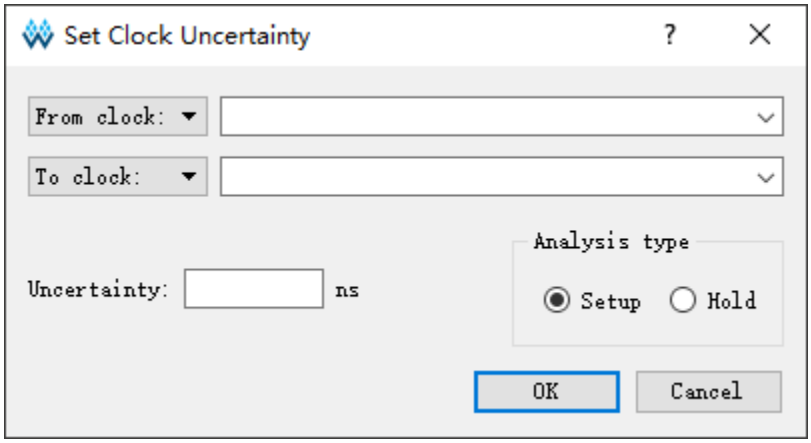
Set Clock Uncertainty

クロックのばらつき値を設定します。

以下は、Clock Uncertainty の新規作成の手順です。

1. 「Constraints」メニューで「Set Clock Uncertainty」を選択すると、「Set Clock Uncertainty」ダイアログボックスがポップアップします(図 3-21)。
   - From clock：送信側のクロックを指定します。右側の「˅」をクリックして選択します。
   - To clock：受信側のクロックを指定します。右側の「˅」をクリックして選択します。
   - Uncertainty：浮動小数点数型、小数点以下 3 桁、単位は ns。クロックのばらつき値を設定します。
   - Analysis type：解析のタイプを示します。

図 3-21 ばらつきの設定



2. 左側のドロップダウン・リストで From のタイプ(From clock、Rise from、Fall from)と To のタイプ(To clock、Rise to、Fall to)を選択し、右側のドロップダウン・リストで現在のすべての作成済み Clock からオブジェクトの Clock を選択します。
3. 情報の入力が完了後、「OK」をクリックして制約を保存すると、Uncertainty の追加が完了します。

Set Clock Group

クロックグループを設定します。

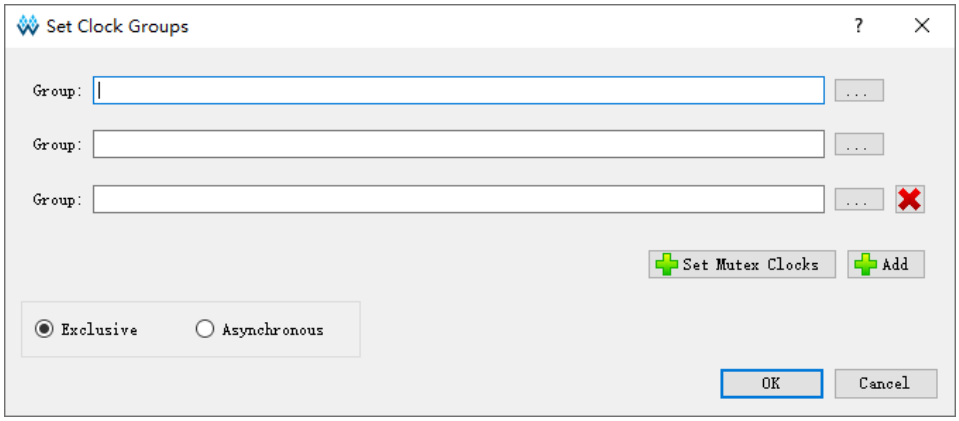
デフォルトでは、Gowin ソフトウェアは、デザイン内のすべてのクロックが同じグループに属し、すべて関連していると想定しています。

この制約文を使用して、クロック間の関係を厳密に指定し、非同期または相互に排他的なクロックに対してクロックグループ制約をかけることをお勧めします。

新しい Clock Group を作成する方法は次のとおりです。

1. 「Constraints」メニューで「Set Clock Groups」を選択すると「Set Clock Groups」ダイアログボックスがポップアップします(図 3-22)。
   - Group：右側の「...」をクリックして選択します。グループ内のクロックを示し、少なくとも 1 つのクロックを指定する必要があります。
   - Set Mutex Clocks：複数のクロックグループを一度に設定するために使用されます。
   - Add：Group 項目をもう 1 つ追加します。
   - Exclusive は、クロックが相互に排他的であり、クロックが同時に有効ではないことを示します。たとえば、Clock0 と Clock1 は MUX2 を経由したあと、1 つのクロックのみが出力されます。
   - Asynchronous は、クロックが非同期であり、クロックのクロックソースが異なることを示します。

図 3-22 クロックグループの設定



2. 「...」ボタンをクリックし、Group の Clock を選択します。追加した Group を削除したい場合、対応する項目の右にある「✖」ボタンをクリックします。
3. 「OK」をクリックして制約を保存します。

注記：

オプション「Exclusive」と「Asynchronous」は、同じ効果を持ちます。

3.7.2 I/O 遅延制約

set_input_delay

データ入力の遅延時間値を設定します。

ユーザーは入力遅延時間値を設定する必要があり、Gowin ソフトウェアは指定された遅延時間値に基づいてタイミング解析を実行します。Gowin ソフトウェアによって生成されたタイミングレポートでは、入力遅延時間のタイプは「tIn」です。

set_output_delay

データ出力の遅延時間値を設定します。

ユーザーは出力遅延時間値を設定する必要があり、Gowin ソフトウェアは指定された遅延時間値に基づいてタイミング解析を実行します。

Gowin ソフトウェアのタイミングレポートでは、出力遅延時間のタイプは「tOut」です。

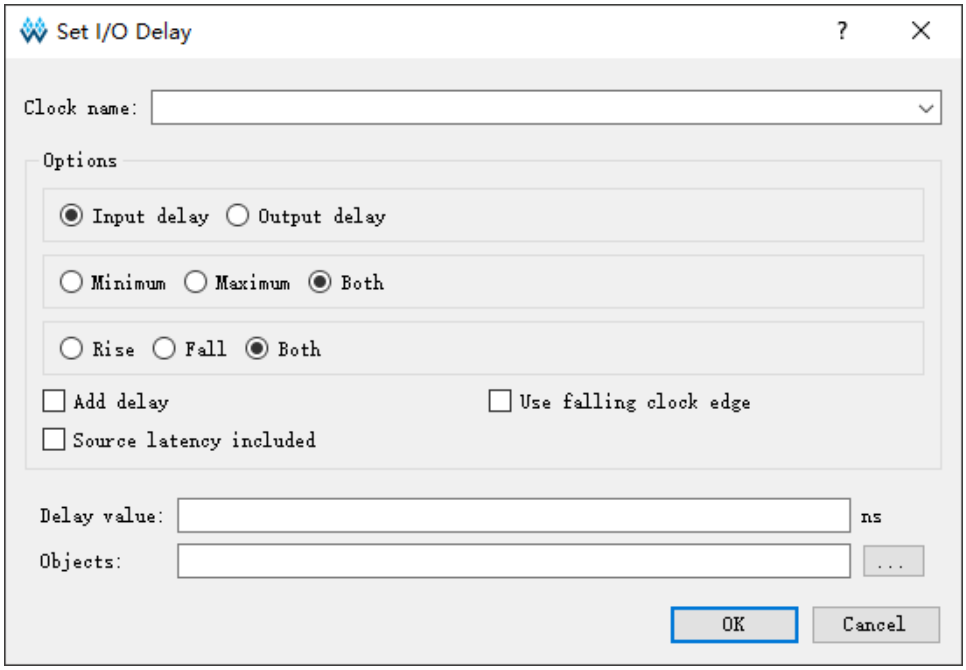
以下は、I/O Delay 制約の新規作成方法です。

1. 「Constraints」メニューで「Set I/O Delay」を選択すると「Set I/O Delay」ダイアログボックスがポップアップします(図 3-23)。
   - Clock name は、I/O に関連付けられたクロックの名前を示します。これは既存のクロックである必要があります。右側の「⌄」をクリックして選択します。
   - Options では、遅延時間タイプ、最大および最小遅延時間、クロックエッジなどを構成することができます。
   - Input delay、Output delay：それぞれ入力遅延時間および出力遅延時間で、相互に排他的です。
   - Minimum、Maximum：I/O の最小遅延時間および最大遅延時間です。Both は、2 つの遅延時間値が同じであることを意味します
   - Rise、Fall：それぞれ立ち上がりエッジ有効と立ち下がり有効を指定します。Both は両者有効を示します。
   - Delay value：I/O の遅延時間値を設定します。浮動小数点数型、小数点以下 3 桁、単位は ns。負の数は早着を意味し、正の数は遅着を意味します。
   - Objects：入出力ポートを指定します。右側の「...」ボタンをクリックして選択してください。
   - Add delay：同じポートに遅延時間値を追加するために使用されます。同じポートに複数の遅延時間値が存在時間する場合、ソフトウェアは setup と recovery の解析に最大値を、hold と removal の

解析に最小値を選択します。このオプションが指定されていない場合、同じポートの同じ制約が上書きされます。

- Use falling clock edge：チェックされている場合、関連クロックの立ち下がりエッジに関連していることを示します。デフォルトは立ち上がりエッジに関連しています。
- Source Latency include：チェックされている場合、設定された遅延時間値にクロックの遅延時間値がすでに含まれていることを意味します。チェックされていない場合、Gowin ソフトウェアは計算にクロックの遅延時間値を入れます。

図 3-23 I/O Delay 制約の作成



2. 構成が完了後、「OK」をクリックして制約を保存します。

3.7.3 例外制約

タイミングタイミング例外を使用することにより、ユーザーは特定のパスのデフォルトの静的タイミング解析ルールを変更できます。set_false_path、set_max_delay、set_min_delay、および set_multicycle_path の 4 つのタイミング例外制約コマンドがあります。

Set False Path

フォルスパスを設定します。設定されたタイミングパスはタイミング解析されません。

以下は、False Path 制約の新規作成方法です。

1. 「Constraints > Set False Path」を選択すると、「Set False Path」ダ

イアログボックスがポップアップします(図 3-24)。

- Analysis type は、解析されないタイプを指定します。Both は、両者とも解析されないことを意味します。
- From はパスの始点を指定します。
- To はパスの終点を指定します。
- Through は、パスの通過点またはネットを指定します。

注記：

From、To、および Through は、単独で使用するか、併用することができます。

図 3-24 False Path 制約の作成

2. 右側の「...」を右クリックし、From、To、および Through に対応する Object を選択します(図 3-14)。

Set Max/Min Delay

最大/最小遅延時間値を設定します。

最大遅延時間が設定されている場合は、setup 解析レポートで報告され、最小遅延時間が設定されている場合は、Hold 解析レポートで報告されます。

以下は、Max/Min Delay 制約の新規作成方法です。

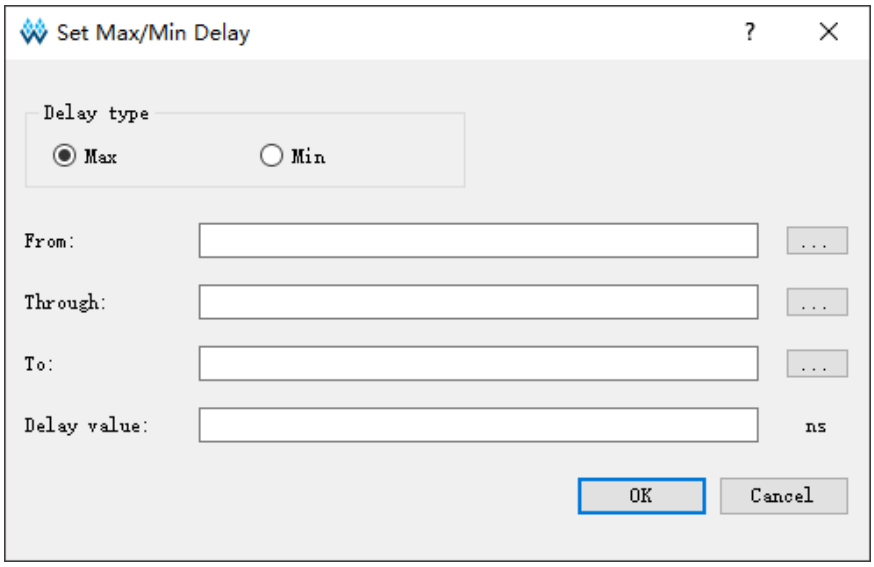
1. 「Constraints>Set Max/Min Delay」を選択すると、「Set Max/Min Delay」ダイアログボックスがポップアップします(図 3-25)。
   - From：パスの始点を指定します。右側の「...」をクリックして選択します。
   - To：パスの終点を指定します。右側の「...」をクリックして選択します。
   - Through：パスの通過点またはネットを指定します。右側の「...」をクリックして選択します。

- Delay value：ユーザーが指定した遅延時間値。浮動小数点数型、小数点以下 3 桁。単位は ns。

注記：

From、To、および Through は、単独で使用するか、併用することができます。

図 3-25 Max/Min Delay 制約の作成



2. 上図に従って構成します。Delay 情報の入力後、「OK」をクリックして作成を完了します。

Set Multicycle Path

マルチサイクルパスを設定します。

デフォルトでは、Gowin ソフトウェアはシングルサイクルのクロック解析を実行します。

以下は、Multicycle Path 制約の新規作成方法です。

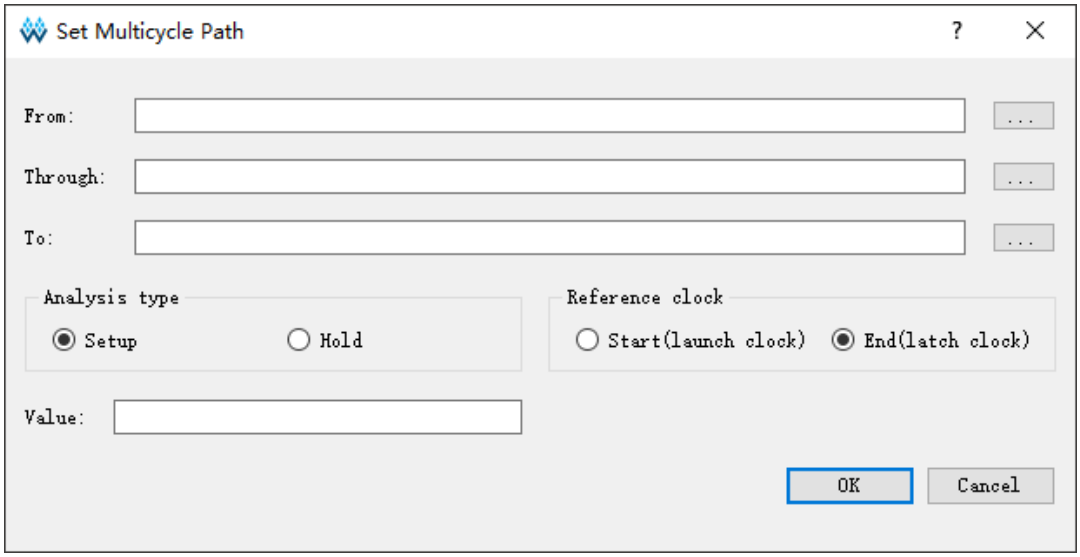
1. 「Constraints>Set Multicycle Path」を選択すると、「Set Multicycle Path」ダイアログボックスがポップアップします(図 3-26)。
   - Reference clock は、リファレンス・クロックが開始クロックかラッチクロックかを指定します。
   - Analysis type は、Setup または Hold のチェックをシチエします。
   - From：パスの始点を指定します。右側の「...」をクリックして選択します。
   - Through：パスの通過点を指定します。右側の「...」をクリックして選択します。
   - To：パスの終点を指定します。右側の「...」をクリックして選択します。

- Value：マルチサイクルの数を指定します。正または負の整数。負の場合は繰り上げを意味し、正の場合は繰り下げを意味します。

注記：

From、To、および Through は、単独で使用するか、併用することができます。

図 3-26 Multicycle Path 制約の作成



2. 構成した後、「OK」をクリックして制約を保存します。

3.7.4 動作条件の制約

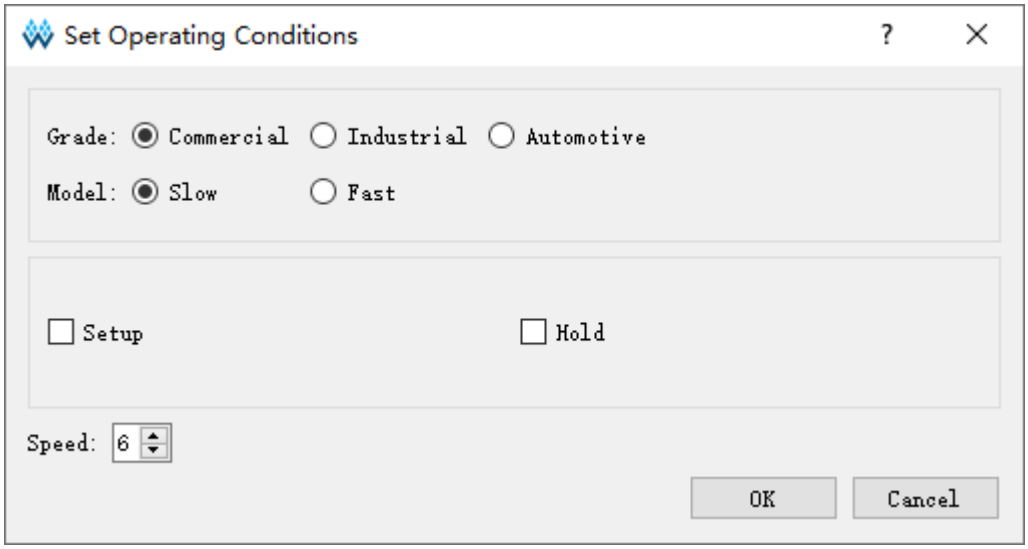
動作条件を設定します。

デフォルトでは、Gowin ソフトウェアは、Setup 解析を実行する時に Slow Model(低速遅延時間モデル)を使用し、Hold 解析を実行する時に Fast Model(高速遅延時間モデル)を使用します。また、ユーザーは、遅延時間モデルをカスタマイズすることもできます。完了後、STA Tool Run Summary で使用されている遅延時間モデルを確認できます。

「Constraints>Set Operating Conditions」を選択すると、「Set Operating Conditions」ダイアログボックスがポップアップします(図 3-27)。

- Grade には、Commercial(コマーシャル)、Industrial(インダストリアル)、Automotive(オートモーティブ)があります。
- Model には、低速と高速があります。
- Hold と Min は hold と removal に有効です。
- Setup と Max は setup と recovery に有効です。
- Max-Min は、setup、hold、recovery、removal に有効です。

図 3-27 Operating Conditions 制約の作成



注記：

- 設定された Grade、Speed がチップの型番と一致しない場合、実際の制約が優先されます。
- 実際の制約の Grade と Speed が現在のプロジェクトをサポートしていない場合、警告メッセージが表示されます。
- setup のみが選択されている場合、hold は、setup の場合の Grade-Speed に従って分析されます。
- hold のみが選択されている場合、setup は、hold の場合の Grade-Speed に従って分析されます。
- GW2A(*)と GW1N(*)シリーズの Slow モデルは高温低圧、Fast モデールは低温高圧です。
- GW5A(*)シリーズの Slow モデルは高温高圧、Fast モデールは低温低圧です。。

3.7.5 レポート制約

Report Timing

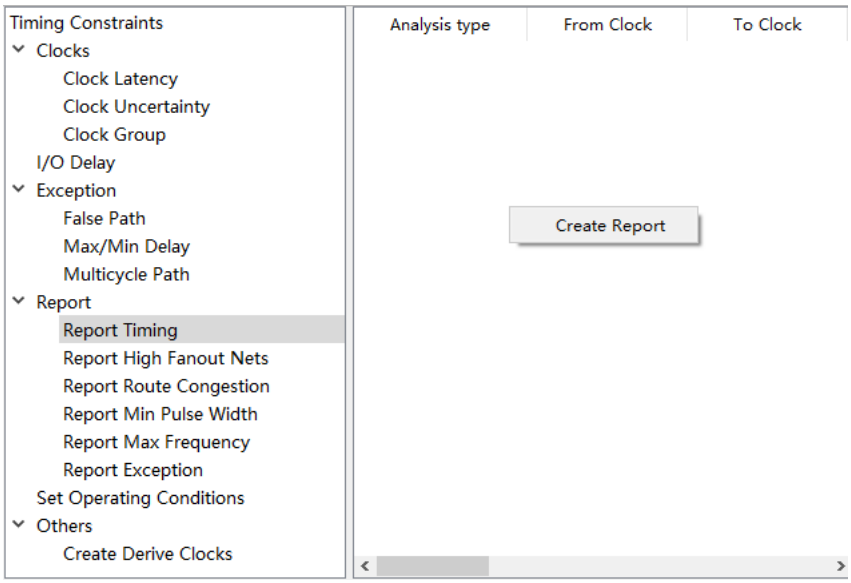
タイミングを報告します。

設定されたパラメータに従って、対応するレポート内容を出力します。これにより、より具体的なタイミングレポートと解析を実現できます。その結果については、setup、hold レポートを参照してください。

その操作手順は以下のとおりです。

1. メインウィンドウで「Timing Constraints > Report Timing」を選択し、空白スペースで右クリックすると、「Create Report」が表示されます(図 3-28)。

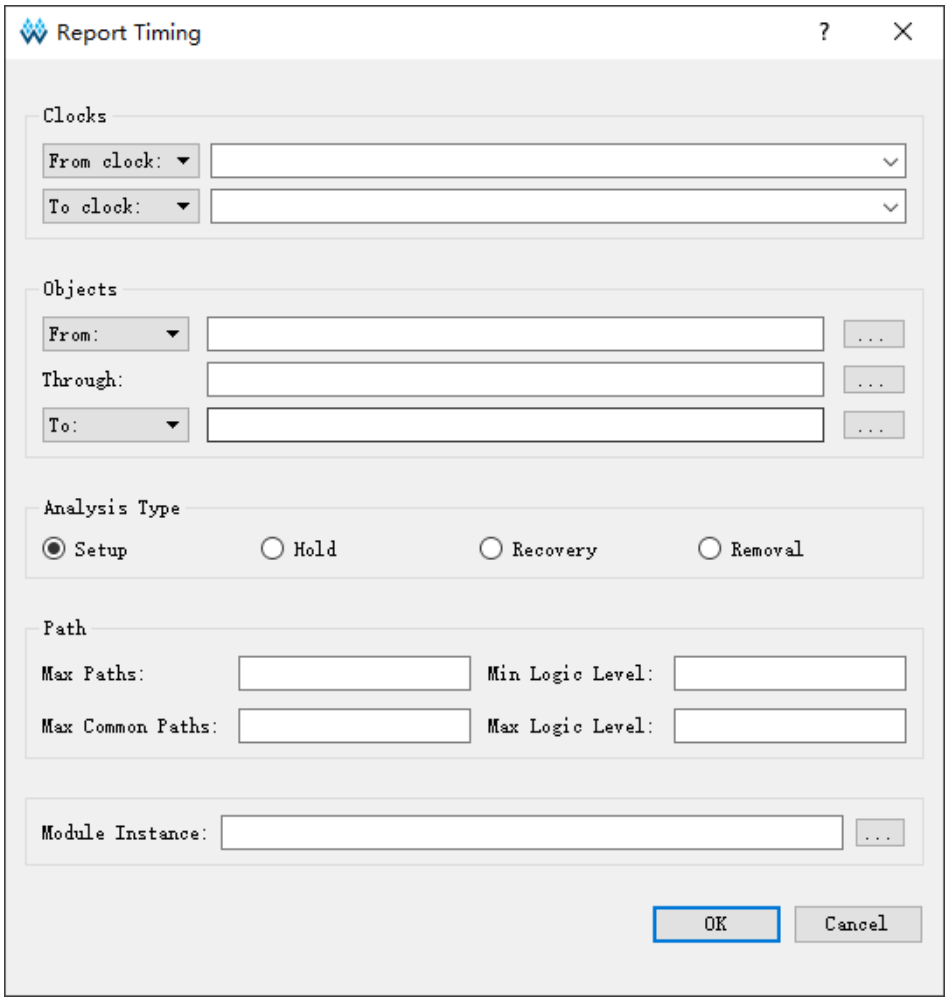
図 3-28 Report Timing の新規作成



2. 「Create Report」を選択すると、図 3-29 に示すダイアログボックスがポップアップします。

- Path は、タイミングレポートの最大パス数(Max Paths)、最大共通パス数(Max Common Paths)、最大および最小ロジックレベル(Max/Min Logic Level)を指定します。
- Clocks は、タイミングレポートパスの関連クロックを指定します。From/To Clock はそれぞれ送信クロックとサンプリングクロックを指定します。右側の「⌄」をクリックして選択します。
- Objects は、解析の開始と終了のオブジェクトを指定します。右側の「...」をクリックして選択します。
- Analysis Type には、セットアップ時間(setup)、ホールド時間(hold)、リカバリ時間(recovery)、およびリムーバル時間(removal)があります。
- Module Instance は、モジュールのインスタンスを指定します。右側の「...」をクリックして選択します。

図 3-29 Report Timing ダイアログボックス



3. 構成した後、「OK」をクリックして保存します。

Report High Fanout Nets

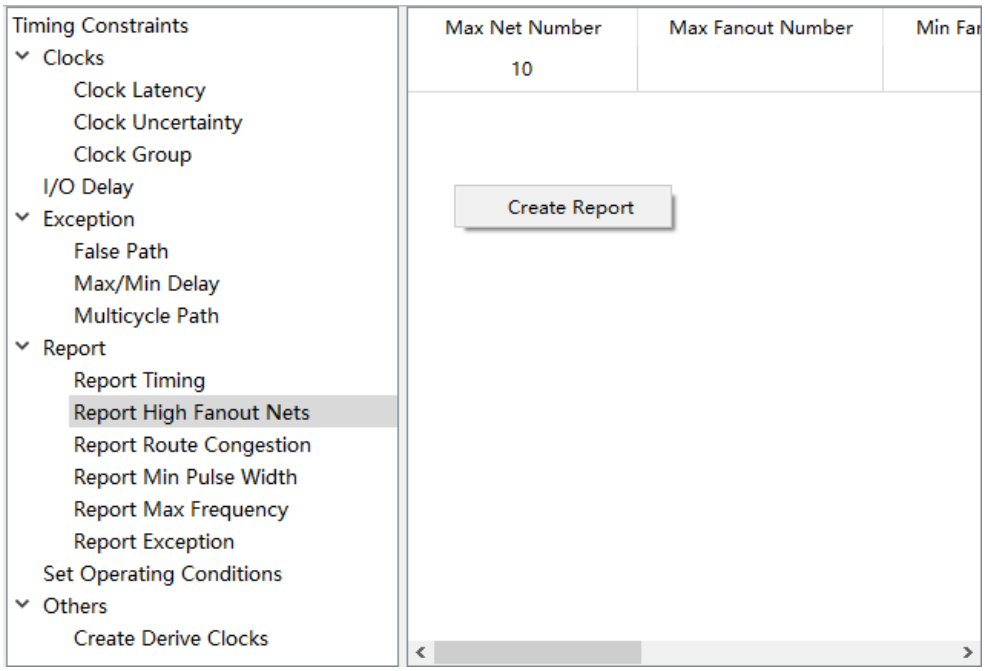
高ファンアウトの net を報告します

デフォルトでは、タイミングパス内の 10 本の最高ファンアウト net が報告されます。生成されたレポートは、High Fanout Nets Report で確認できます。

その操作手順は以下のとおりです。

1. メインウィンドウで「Timing Constraints>Report High Fanout Nets」を選択します。
2. 右側の空白で右クリックすると、「Create Report」が表示されます(図 3-30)。

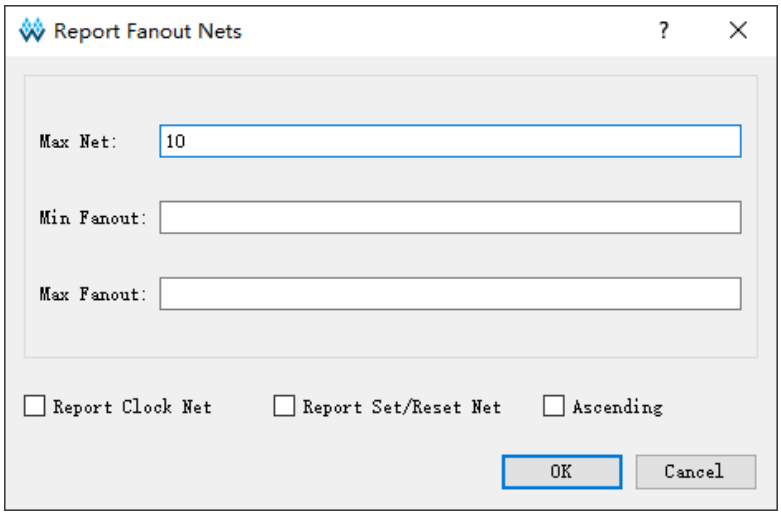
図 3-30 Report High Fanout Nets の作成



3. 「Create Report」を選択すると、図 3-31 に示すダイアログボックスがポップアップします。

- Max Net は、レポートの最大数を指定します。
- Min Fanout と Max Fanout は、それぞれファンアウトの下限と上限を指定します。正の整数です。
- Report Clock Net は、クロック入力に接続されている Net を報告します。
- Report Set/Reset Net は、LSR 入力に接続されている Net を報告します。
- Ascending は Net を昇順でソートするを指定します。デフォルトでは降順です。

図 3-31 Report High Fanout Nets ダイアログボックス



4. 構成した後、「OK」をクリックして保存します。

Report Route Congestion

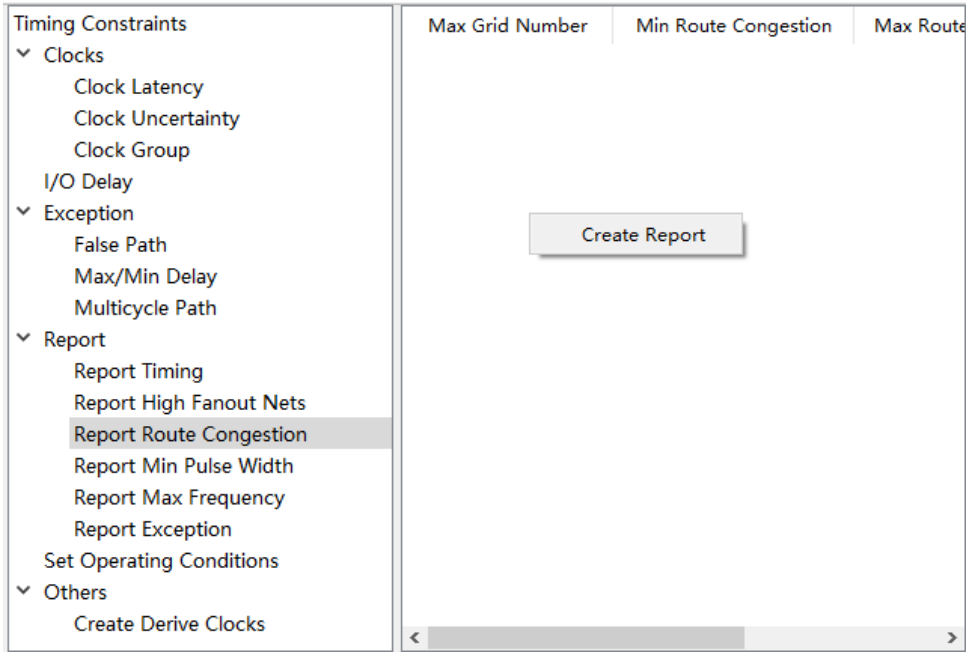
配線の密集レベルを報告します。

デフォルトでは、最悪の 10 個の Grid が報告されます。これは通常、特定の Grid の配線の密集レベルの報告に使用されます。生成されたレポートは、Route Congestions Report で確認できます。

その手順は以下のとおりです。

1. メインウィンドウで、「Timing Constraints>Report Route Congestion」を選択します。
2. 右側の空白で右クリックすると、「Create Report」が表示されます(図 3-32)。

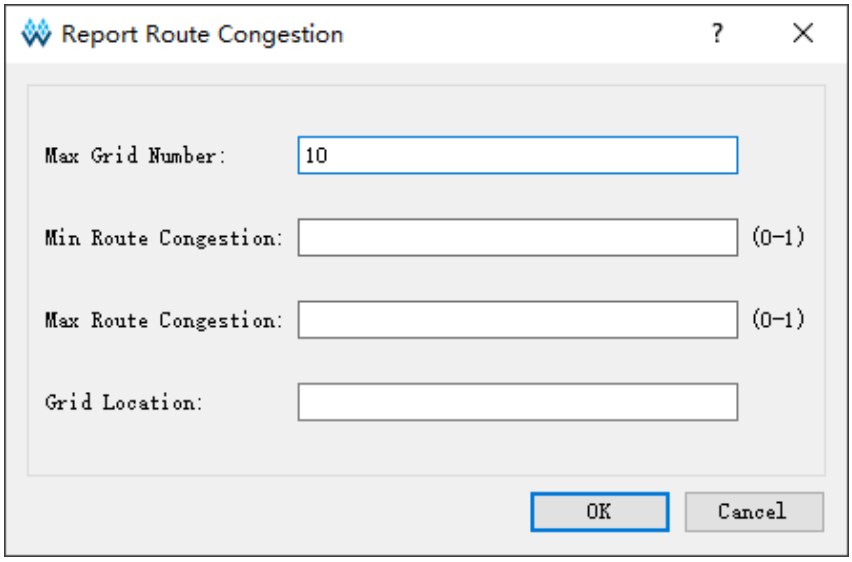
図 3-32 Report Route Congestion の作成



3. 「Create Report」を選択すると、図 3-33 に示すダイアログボックスがポップアップします。

- Max Grid Number はレポートの数を指定します。
- Min、Max Route Congestion は、それぞれ配線の密集レベルの下限と上限を指定します。浮動小数点数型、小数点以下 3 桁。
- Grid Location は、報告される Grid を指定します(例えば、R4C4)。

図 3-33 Report Route Congestion ダイアログボックス



4. 構成した後、「OK」をクリックして保存します。

Report Min Pulse Width

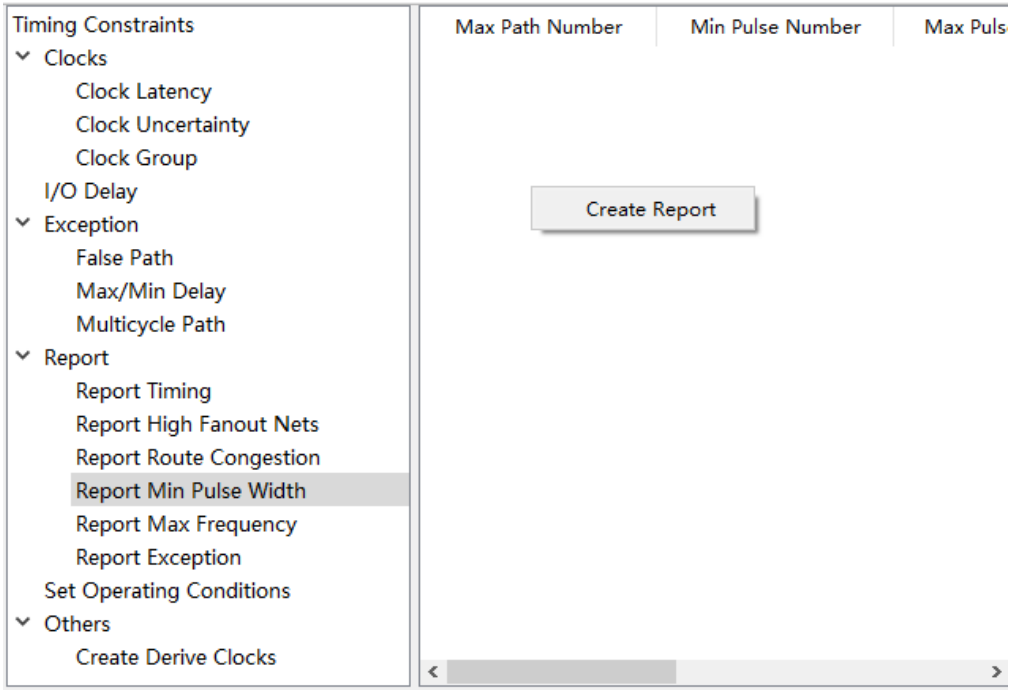
最小パルス幅を報告します。

デフォルトでは、10 個が報告されます。ユーザーはこの制約文を使用して、特定の範囲内のパルス幅または特定のオブジェクトでのパルス幅を報告できます。生成されたレポートは、Minimum Pulse Width Report で確認できます。

手順は以下のとおりです。

1. メインウィンドウで「Timing Constraints>Report Min Pulse Width」を選択します。
2. 右側の空白で右クリックすると、「Create Report」が表示されます(図 3-34)。

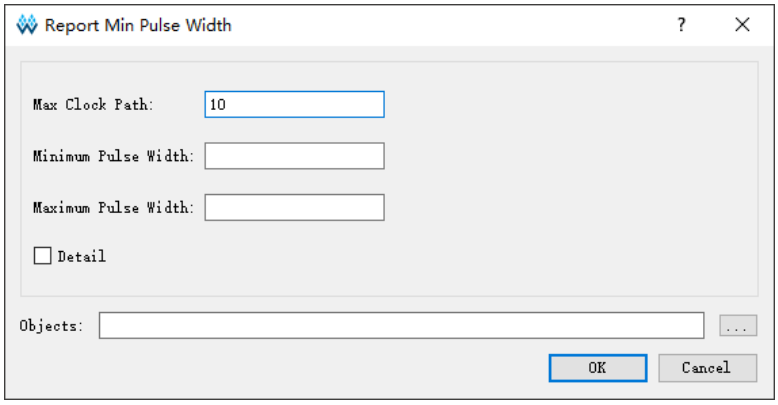
図 3-34 Report Min Pulse Width の作成



3. 「Create Report」を選択すると、図 3-35 に示すダイアログボックスがポップアップします。
   - Max Clock Path は、レポートの最大数を指定します。正の整数です。
   - Minimum、Maximum Pulse Width は、それぞれ報告されるパルス幅の下限と上限を指定します。浮動小数点数型、小数点以下 3 桁。
   - Detail は、詳細なパスを報告するかどうかを指定します。
   - Objects は、報告する必要のあるシーケンシャル・エレメントを

指定します。DFF などの FF のみがサポートされます。右側の「...」をクリックして選択します。

図 3-35 Report Min Pulse Width ダイアログボックス



4. 構成した後、「OK」をクリックして保存します。

Report Max Frequency

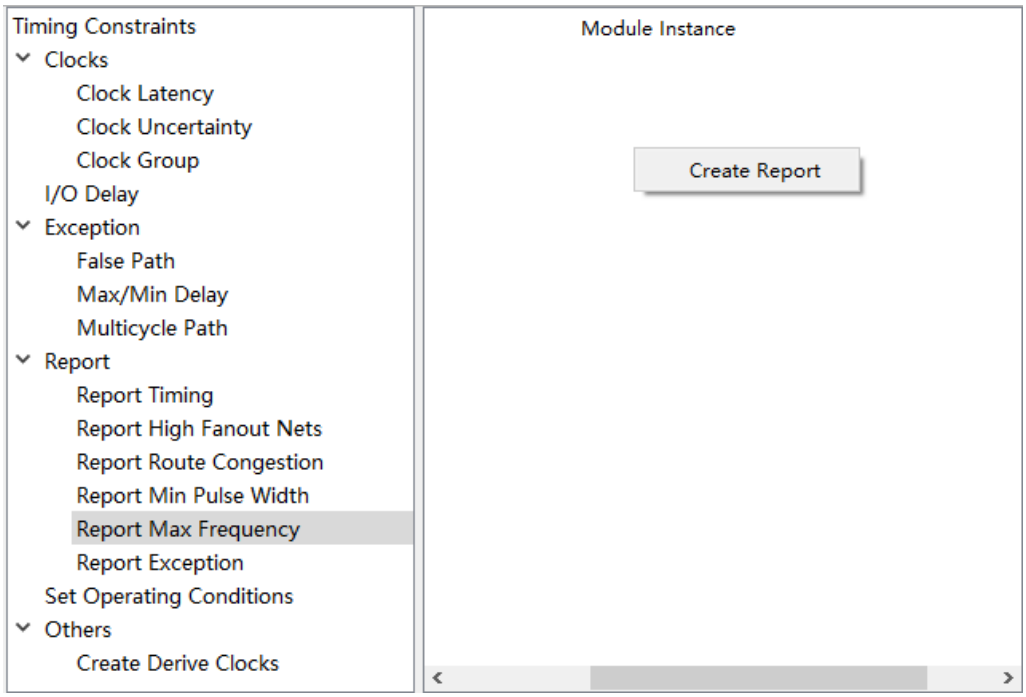
最大動作周波数を報告します。

デフォルトでは、Gowin ソフトウェアは Top 層のクロックの最大周波数のみを報告します。ユーザーは、特定のモジュールの最大動作クロック周波数のレポートを指定できます。 このモジュールの最大動作クロック周波数のクリティカルパスは、このモジュール内に限定されないが、このモジュールのタイミング終点に関連します。

その操作手順は以下のとおりです。

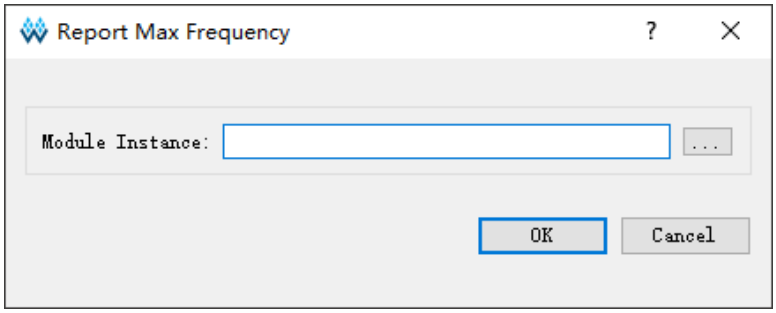
1. メインウィンドウで「Timing Constraints > Report > Report Max Frequency」を選択します。
2. 右側の空白で右クリックすると、「Create Report」が表示されます(図 3-36)。

図 3-36 Report Max Frequency の作成



3. 「Create Report」を選択すると、図 3-37 に示すダイアログボックスがポップアップします。「Module Instance」はモジュールのインスタンスの名前です。右側の「...」をクリックして選択します。

図 3-37 Report Max Frequency ダイアログボックス



4. 構成した後、「OK」をクリックして保存します。

Report Exception

例外制約を報告します。

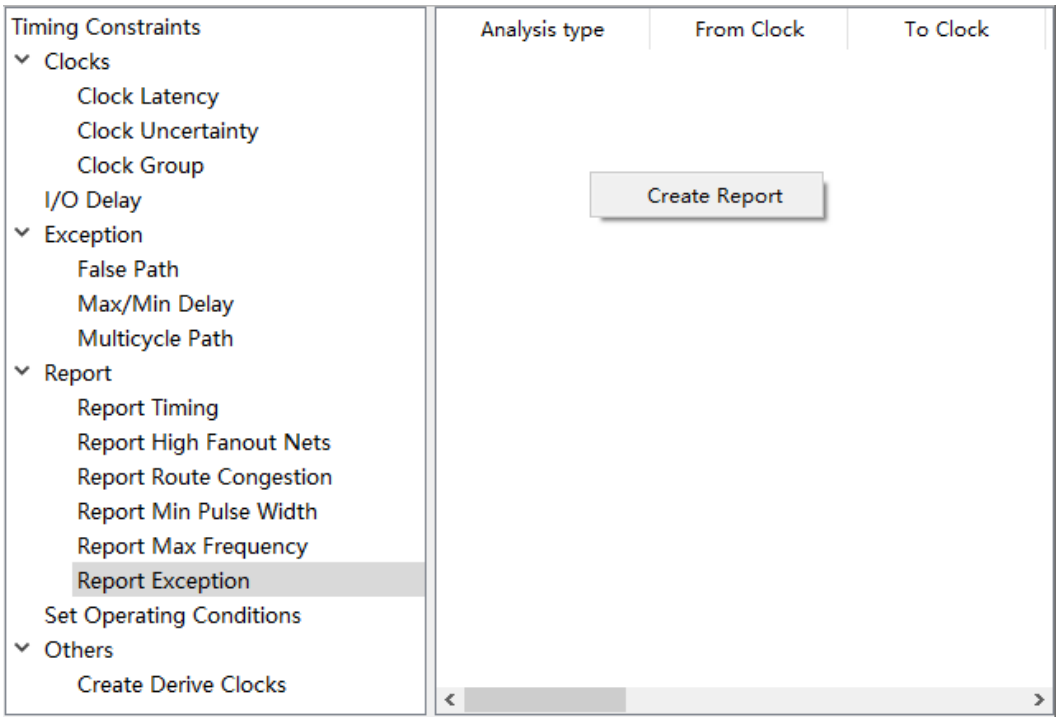
例外制約文の影響を受けるタイミングパスを制約して、ユーザーが関心のあるタイミングパスを報告します。

その手順は以下のとおりです。

1. メインウィンドウで「Timing Constraints > Report > Report Exception」を選択します。
2. 右側の空白で右クリックすると、「Create Report」が表示されます(図

3-38)。

図 3-38 Report Exception の作成

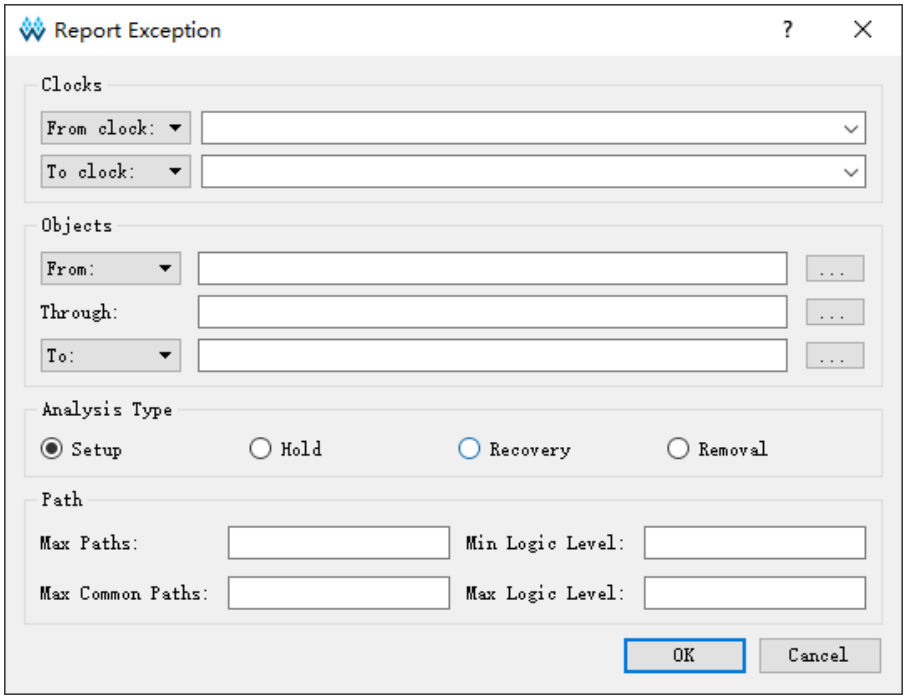


3. 「Create Report」を選択すると、図 3-39 に示すダイアログボックスがポップアップします。

注記：

各オプションの詳細については、Report Timing を参照してください。

図 3-39 Report Exception ダイアログボックス



4. 構成した後、「OK」をクリックして保存します。

3.7.6 その他の制約

Create Derive Clocks

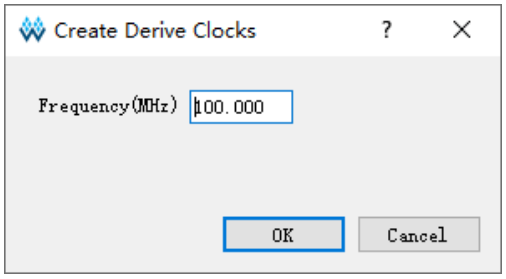
グローバルのクロック周波数を指定します。最大 1200MHz の周波数をサポートします。

Derive Clocks 制約を追加するには、2 つの方法があります。

1. 「Constraints」メニューで Derive Clocks 制約を追加します。

   a). 「Constraints > Create Derive Clocks…」を選択すると「Create Derive Clocks」ダイアログボックスがポップアップします(図 3-40)。

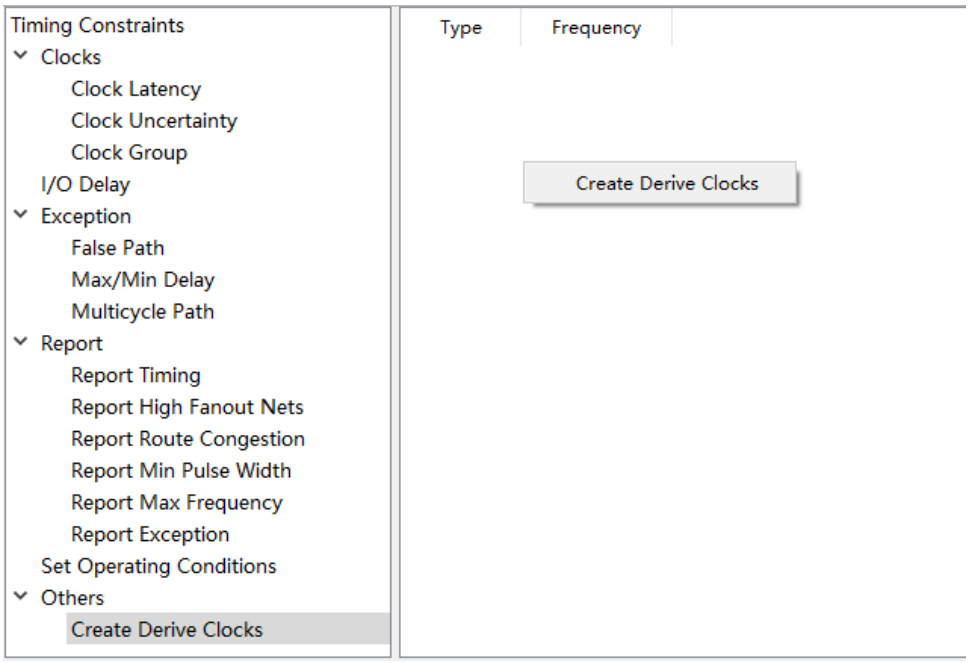
図 3-40 Derive Clocks の作成



   b). Frequency(MHz)：グローバルの周波数、1200 以下の正の浮動小数点数、小数点以下 3 桁。

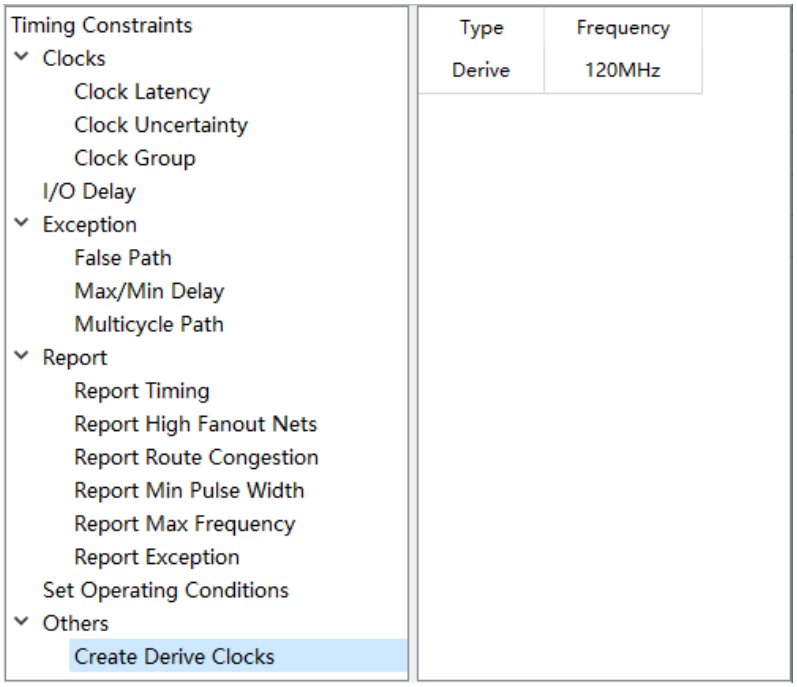
2. 「Others > Create Derive Clocks」で Derive Clocks を作成します。空白部分を右クリックし、「Create Derive Clocks」を選択して Derive Clocks を作成します(図 3-41)。

図 3-41 Create Derive Clocks を選択



クロックが作成されると、Derive Clocks リストには対応する制約が追加されます(図 3-42)。

図 3-42 Derive Clocks リスト



3.7.7 保存とエクスポート

すべての制約の編集が完了後、「File>Save」または「File>Save As」をクリックすると、現在のタイミング制約エディタの制約情報がタイミン

グ制約ファイル(.sdc)に保存されます。

3.8 タイミング制約の優先度

Gowin ソフトウェアにより提供されるタイミング制約の優先度は次に示すとおりです(低い順)。

1. create_clock と create_generated_clock。
2. derive_clocks；
3. set_multicycle_path。
4. set_max_delay と set_min_delay。
5. set_false_path。
6. set_clock_groups。

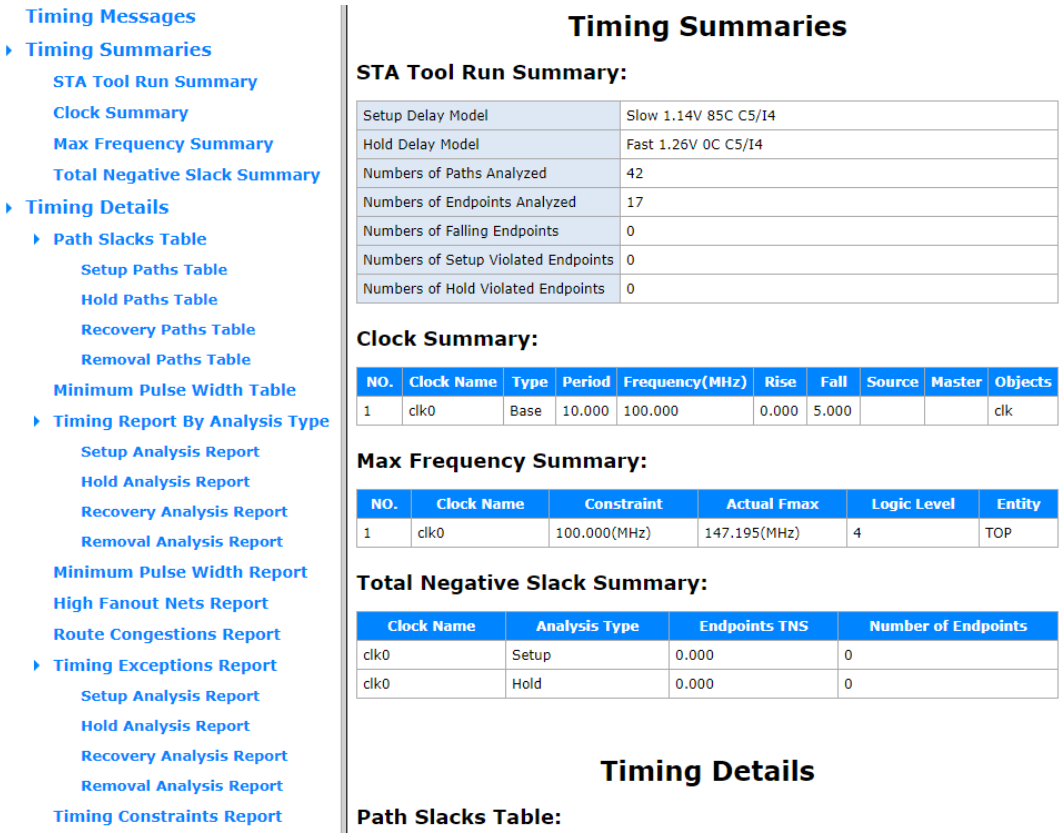
注記：

同じタイミングパスで競合が生じる可能性のあるタイミング制約のみが優先度で並べ替えられます。言及されていない他の制約の場合は、異なるタイプの制約間の競合が生じません。

4 タイミングレポート

このセクションでは、GOWIN セミコンダクターの静的タイミング解析のレポートについて説明します。図 4-1 に示すように、レポートは左側のナビゲーションバーと右側のコンテンツバーからなり、要件を満たさない項目は赤で表示されます。

図 4-1 静的タイミング解析レポート



4.1 Timing Summaries

タイミングサマリ(Timing Summaries)は、STA Tool Run Summary、Clock Summary、Max Frequency Summary、および Total Negative Slack

Summary の 4 つの部分で構成されています(図 4-2)。

図 4-2 Timing Summaries

Timing Summaries

STA Tool Run Summary:

Setup Delay Model	Slow 1.14V 85C C5/I4
Hold Delay Model	Fast 1.26V 0C C5/I4
Numbers of Paths Analyzed	42
Numbers of Endpoints Analyzed	17
Numbers of Falling Endpoints	0
Numbers of Setup Violated Endpoints	0
Numbers of Hold Violated Endpoints	0

Clock Summary:

NO.	Clock Name	Type	Period	Frequency(MHz)	Rise	Fall	Source	Master	Objects
1	clk0	Base	10.000	100.000	0.000	5.000			clk

Max Frequency Summary:

NO.	Clock Name	Constraint	Actual Fmax	Logic Level	Entity
1	clk0	100.000(MHz)	147.195(MHz)	4	TOP

Total Negative Slack Summary:

Clock Name	Analysis Type	Endpoints TNS	Number of Endpoints
clk0	Setup	0.000	0
clk0	Hold	0.000	0

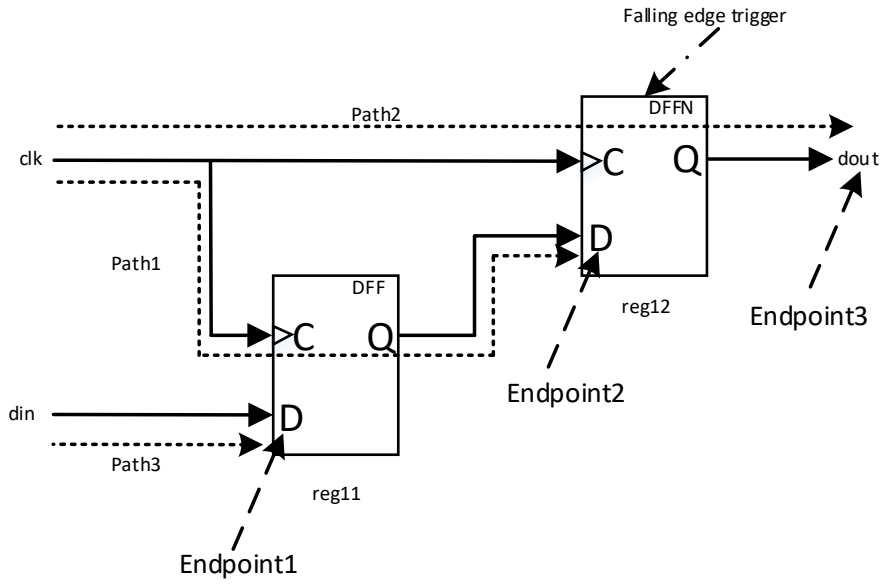
4.1.1 STA Tool Run Summary

- Setup Delay Model：セットアップ時間の解析のために使用されるデータモデル。デフォルトでは、Slow モデルが使用されます。
- Hold Delay Model：ホールド時間の解析のために使用されるデータモデル。デフォルトでは、Fast モデルが使用されます。
- Numbers of Paths Analyzed：静的タイミング解析パスの数。図 4-3 に示すように、Path1、Path2、および Path3 という合計 3 つのタイミングパスが解析されました。
- Numbers of Endpoints Analyzed：解析されたタイミングパス終点の数。図 4-3 に示すように、Endpoint1、Endpoint2、Endpoint3 という合計 3 つの終点が解析されました。
- Numbers of Falling Endpoints：立ち下がりエッジ・トリガの終点の数。図 4-3 に示すように、reg12 は立ち下がりエッジ・トリガのフリ

ップフロップであるため、終点 D は立ち下がりエッジ・トリガの終点となります。

- Numbers of Setup Violated Endpoints：セットアップ時間を満たさない終点の数。
- Numbers of Hold Violated Endpoints：ホールド時間を満たさない終点の数。

図 4-3 Path & Endpoints



4.1.2 Clock Summary

ユーザーデザインのすべてのクロックを報告します。デザイン内のクロックが制約されていない場合、ソフトウェアはデフォルトのクロックを作成します。Arora ファミリーの場合、クロック周波数はデフォルトで 100MHz、LittleBee ファミリーの場合、クロック周波数はデフォルトで 50MHz です。

- NO.：番号。
- Clock Name：クロックの名前。;
- Type：Base と Generated の 2 つのタイプがあります。Base は基本クロックを表し、Generated は派生クロックを表します。
- Period：クロックサイクル、単位は ns。
- Frequency (MHz) ：クロック周波数。
- Rise：クロックの立ち上がりエッジ時間。
- Fall：クロックの立ち下がりエッジ時間。
- Source：クロックソース。port、pin、net、reg から取得できます。

- Master：マスタークロック。
- Objects：port、pin、net、reg などのクロックオブジェクト。

注記：

- デフォルトでクロックを作成する際、クロック名が重複している場合、Gowin ソフトウェアは自動的にサフィックス「_gowin」を追加します。
- PLL、OSC、および CLKDIV タイプの場合、Gowin ソフトウェアはデフォルトでクロック名にサフィックス「.default_gen_clk」を追加します。
- ソフトウェアは PLL クロック出力ポート間の関係を自動的に解析します。例えば、CLKOUTP は CLKOUT に基づいて位相とデューティサイクルが調整されたクロック出力信号です。

4.1.3 Max Frequency Summary

- NO.：番号。
- Clock Name：タイミングモデルを駆動するクロックの名前。
- Constraint：SDC 制約のクロック周波数、または SDC 制約なしの場合のデフォルトのクロック周波数。
- Actual Fmax：配置配線の後に Gowin ソフトウェアの分析による最大の実際の周波数。
- Logic Level：ロジックレベル数。
- Entity：最大周波数のモジュール。デフォルトは TOP です。

注記：

- 配置配線の後にクロックがタイミングモデルを駆動しない場合は、「No timing paths to get frequency of *」が表示されます。
- 最大クロック周波数レオートでは、同期クロックのみが報告されます。
- Gowin ソフトウェアがより正確な解析を実行できるように、デザインに完全なタイミング制約を追加することをお勧めします。

4.1.4 Total Negative Slack Summary

- Clock Name：クロックの名前。
- Analysis Type：setup と hold の 2 種類があります。
- Endpoints TNS：クロック(ClockName に対応)によって駆動されるタイミングパスの終点スラックが負の場合の合計時間。同じ終点を持つパスに対しては、最悪のパスのみが集計されます。
- Number of Endpoints：クロック(ClockName に対応)によって駆動されるタイミングパスの終点スラックが負の場合の合計終点数。同じ終点を持つパスに対しては、最悪のパスのみが集計されます。

4.2 Timing Details

4.2.1 Path Slacks Table

Path Slacks Table は、Setup Paths Table(セットアップ時間パス解析テーブル)、Hold Paths Table(ホールド時間パス解析テーブル)、Recovery Paths Table(リカバリー時間パス解析テーブル)、Removal Paths Table(リムーバル時間パス解析テーブル)で構成された、タイミングパスの静的解析のスラックテーブルです。上記の 4 種類のテーブルのヘッダー(図 4-4 参照)の説明は次のとおりです。

- Path Number：パス番号。デフォルトの最大レポート数は 25 です。
- Path Slack：データに所要時間からデータ到着時間を引いた値であり、この値が負の場合、タイミングは満たされません。
- From Node：前段シーケンシャル・エレメントのタイミング解析の開始ノード。
- To Node：後段のシーケンシャル・エレメントのタイミング解析の終了ノード。
- From Clock：前段シーケンシャル・エレメントのデータ送信クロックと送信エッジ・タイプ(立ち上がりエッジまたは立ち下がりエッジ)。
- To Clock：後段のシーケンシャル・エレメントのデータ・ラッチ・クロックとラッチ・エッジ・タイプ。
- Relation：送信クロックとサンプリングクロックの時間関係を示します。
- Clock Skew：クロックスキュー。送信クロックとラッチクロックが前段と後段のシーケンシャル・エレメントに到着する時間差を指します。
- Data Delay：データ到着パスにおけるデータ遅延時間です。その値は、データ到着パス全体における遅延時間値の一部です。

注記：

- 解析に使用できるタイミングパスがない場合、「Nothing to report!」と表示されます。
- Path Slacks Table では、デフォルトで最悪の 25 パスが出力されます。ユーザーは Report Timing を利用してこの 25 パス以外のパスを確認できます。
- Path Slacks Table には、デフォルトでクロック・ドメイン・クロッシングのタイミングパスが含まれます。クロック・ドメイン・クロッシングの解析が不要な場合は、set_clock_groups または set_false_path で指定できます。

図 4-4 Path Slacks Table

Path Slacks Table:

Setup Paths Table

Report Command:report_timing -setup -max_paths 25 -max_common_paths 1

Path Number	Path Slack	From Node	To Node	From Clock	To Clock	Relation	Clock Skew	Data Delay
1	8.806	synS_r_s0/Q	synE_r_s0/D	ck0:[R]	ck0:[R]	10.000	0.000	0.794

Hold Paths Table

Report Command:report_timing -hold -max_paths 25 -max_common_paths 1

Path Number	Path Slack	From Node	To Node	From Clock	To Clock	Relation	Clock Skew	Data Delay
1	0.570	synS_r_s0/Q	synE_r_s0/D	ck0:[R]	ck0:[R]	0.000	0.000	0.570

Recovery Paths Table

Report Command:report_timing -recovery -max_paths 25 -max_common_paths 1

Path Number	Path Slack	From Node	To Node	From Clock	To Clock	Relation	Clock Skew	Data Delay
1	8.649	rstSrc_r_s0/Q	rstObj_r_s0/CLEAR	ck0:[R]	ck1:[R]	10.000	0.000	1.278

Removal Paths Table

Report Command:report_timing -removal -max_paths 25 -max_common_paths 1

Path Number	Path Slack	From Node	To Node	From Clock	To Clock	Relation	Clock Skew	Data Delay
1	0.788	rstSrc_r_s0/Q	rstObj_r_s0/CLEAR	ck0:[R]	ck1:[R]	0.000	0.000	0.833

4.2.2 Minimum Pulse Width Table

シーケンシャル・エレメントが認識できる最小パルス幅の静的タイミング解析テーブルです。パルス幅とは、有効な High/Low レベルの信号が持続する時間を指します。デフォルトで最悪の 10 件が報告されます。図 4-5 のヘッダーについて説明します。

- Number：番号。デフォルトは 10 個報告されます。
- Slack：シーケンシャル・エレメントが認識できる最小パルス幅のスラック値。
- Actual Width：配置配線後の静的タイミング解析後にシーケンシャル・エレメントが認識できる実際のパルス幅。
- Required Width：シーケンシャル・エレメントが認識できるために必要な最小パルス幅。
- Type：パルス幅のタイプ。Low Pulse Width(Low パルス幅)と High Pulse Width(High パルス幅)の 2 つのタイプがあります。
- Clock：最小パルス幅解析用のクロック。
- Objects：最小パルス幅解析用のシーケンシャル・エレメントのインスタンス。

注記：

最小パルス幅解析レポートがない場合、「Nothing to report!」と表示されます。

図 4-5 Minimum Pulse Width Table

Minimum Pulse Width Table:

Report Command:report_min_pulse_width -nworst 10 -detail

Number	Slack	Actual Width	Required Width	Type	Clock	Objects
1	2.738	4.238	1.500	Low Pulse Width	DEFAULT_CLK	reg12
2	2.738	4.238	1.500	Low Pulse Width	DEFAULT_CLK	reg11_Z
3	2.813	4.313	1.500	High Pulse Width	DEFAULT_CLK	reg12
4	2.813	4.313	1.500	High Pulse Width	DEFAULT_CLK	reg11_Z

4.2.3 Timing Report By Analysis Type

Setup Analysis Report、Hold Analysis Report、Recovery Analysis Report、および Removal Analysis Report の 4 つのタイプの静的タイミング解析があります。

Setup Analysis Report

シーケンシャル・エレメントのクロック信号の立ち上がりエッジの前にデータが安定している時間を解析するためのセットアップ時間解析レポートです。この時間が十分でない場合、データはクロックの立ち上がりエッジでシーケンシャル・エレメントに安定的に供給されません。

Gowin ソフトウェアは、詳細な計算と解析を経て、データ到着時間、データ所要時間、サンプリングクロック、送信クロックなどをユーザー参照用の Setup Analysis Report に出力します。

このレポートは、コマンド report_timing -setup によって生成されます。Gowin ソフトウェアは、デフォルトで 25 の最悪のスラックのタイミングパスを分析して報告します。レポートには、Path Summary、Data Arrival Path、および Path Statistics が含まれます。

1. Path Summary。図 4-6 は静的タイミング解析のパス情報の概要です。その詳細は次のとおりです。
   - Slack：許容されるデータ最大遅延時間から実際のデータ到着時間を差し引いた時間です。正の値はタイミング収束を示し、負の値はタイミングが収束しないことを示します。
   - Data Arrival Time：データ到着時間。
   - Data Required Time：データ所要時間。
   - From：前段のシーケンシャル・エレメント。
   - To：後段のシーケンシャル・エレメント。
   - Launch Clock：送信クロック。有効エッジは、R(立ち上がりエッジ)または F(立ち下がりエッジ)です。
   - Latch Clock：サンプリングクロック。有効エッジは、R(立ち上がりエッジ)または F(立ち下がりエッジ)です。

図 4-6 Path Summary

Path Summary:

Slack	5.789
Data Arrival Time	6.767
Data Required Time	12.556
From	reg11_Z
To	reg12_Z
Launch Clk	sysclk1:[R]
Latch Clk	sysclk1:[R]

2. Data Arrival Path。図 4-7 にはデータ到着パスを示します。その詳細は次のとおりです。
   - AT：タイミングパス上の時間ポイントを指します。
   - DELAY：遅延時間値を指します。
   - TYPE：タイミング解析パス上のノードのタイプを示し、空白は該当なしを示します。
   - RF：現在解析されているエレメントの信号遷移のタイプを指します。
   - FANOUT：ファンアウト。
   - LOC：現在解析されているエレメントの、デバイス内の物理的な位置。位置情報なしの場合は UNPLACE マークを使用します。
   - NODE：静的タイミング解析のパス上のノード。インスタンス名とポート、クロック、およびクロックエッジのアクティブ化時間(active clock edge time)が含まれます。

注記：

図 4-7 の各 TYPE の意味は次のとおりです。

- tCL：time of clock latency、クロックソース遅延時間。
- tINS：time of module instance、モジュール遅延時間。
- tNET：time of net、net の遅延時間。
- tIn：time of input、データ入力遅延時間。
- tC2Q：time of clock to output、シーケンシャル・エレメントのクロックから出力までの遅延時間。

図 4-7 Data Arrival Path

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysclk1
0.000	0.000	tCL	RR	1	IOL7[A]	clk1_ibuf/I
0.943	0.943	tINS	RR	2	IOL7[A]	clk1_ibuf/O
3.236	2.293	tNET	RR	1	IOL2[B]	reg11_Z/CLK
3.786	0.550	tC2Q	RF	1	IOL2[B]	**reg11_Z/Q**
6.767	2.981	tNET	FF	1	R5C9[1][A]	**reg12_Z/D**

3. Data Required Path。図 4-8 に示すように、データ所要パスとは、クロックが有効なエッジからシーケンシャル・エレメントのクロックポートに到着するまでのパスを指します。

注記：

図 4-8 の TYPE の意味は次のとおりです。

- tUnc：time of clock uncertainty、クロックのばらつき値。
- tOut：time of output、データ出力遅延時間。
- tSu：time of setup、セットアップ時間。

図 4-8 Data Required Path

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
10.000	10.000					active clock edge time
10.000	0.000					sysclk1
10.000	0.000	tCL	RR	1	IOL7[A]	clk1_ibuf/I
10.943	0.943	tINS	RR	2	IOL7[A]	clk1_ibuf/O
13.236	2.293	tNET	RR	1	R5C9[1][A]	reg12_Z/CLK
13.036	-0.200	tUnc				
12.556	-0.480	tSu		1	R5C9[1][A]	reg12_Z

4. Path Statistics。図 4-9 にパスの統計情報を示します。
   - Clock Skew：クロックスキュー。
   - Setup Relationship：前段のシーケンシャル・エレメントのデータ送信と後段のシーケンシャル・エレメントのデータラッチ間の時間関係。
   - Logic Level：ロジックレベル数。
   - Arrival Clock Path Delay：Data Arrival Path のクロック遅延時間状況の統計を取ります。cell は論理エレメントの遅延時間、route は配線の遅延時間、tC2Q はシーケンシャル・エレメントの内部遅延時間を表します。
   - Arrival Data Path Delay：Data Arrival Path のデータ遅延時間状況

の統計を取ります。

- Required Clock Path Delay：Data Required Path のクロック遅延時間状況の統計を取ります。

図 4-9 Path Statistics

Path Statistics:

Clock Skew	0.000
Setup Relationship	10.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 2.981, 84.423%; tC2Q: 0.550, 15.577%
Required Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%

Hold Analysis Report

図 4-10 はシーケンシャル・エレメントのクロック信号の立ち上がりエッジの前にデータが安定している時間を解析するためのホールド時間解析レポートです。この時間が十分でない場合、データはシーケンシャル・エレメントに安定して供給されません。Gowin ソフトウェアは詳細な計算と分析を実行し、最終的にデータ到着時間、データ所要時間、サンプリングクロック、送信クロックなどをユーザー参照用のレポートに出力します。このレポートは、コマンド report_timing -hold によって生成されます。Gowin ソフトウェアは、デフォルトで 25 の最悪のスラックのタイミングパスを分析して報告します。レポートのヘッダーの詳細については、Setup Analysis Report を参照してください。

図 4-10 Hold Analysis Report

Hold Analysis Report

Report Command:report_timing -hold -max_paths 25 -max_common_paths 1

Path1

Path Summary:

Slack	1.003
Data Arrival Time	3.554
Data Required Time	2.551
From	reg11_s0
To	reg12_s0
Launch Clk	sysyclk:[R]
Latch Clk	sysyclk:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.811	0.811	tINS	RR	2	IOL11[A]	clk_ibuf/O
2.533	1.723	tNET	RR	1	R2C9[0][A]	reg11_s0/CLK
2.933	0.400	tC2Q	RR	1	R2C9[0][A]	**reg11_s0/Q**
3.554	0.621	tNET	RR	1	R2C9[1][A]	**reg12_s0/CLEAR**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.811	0.811	tINS	RR	2	IOL11[A]	clk_ibuf/O
2.533	1.723	tNET	RR	1	R2C9[1][A]	reg12_s0/CLK
2.533	0.000	tUnc				
2.551	0.018	tHld		1	R2C9[1][A]	reg12_s0

Path Statistics:

Clock Skew	0.000
Hold Relationship	0.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.811, 31.998%; route: 1.723, 68.002%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 0.621, 60.818%; tC2Q: 0.400, 39.182%
Required Clock Path Delay	cell: 0.811, 31.998%; route: 1.723, 68.002%

Recovery Analysis Report

図 4-11 はリカバリ時間解析レポートです。リカバリ時間：シーケンシャル・エレメントの有効クロックエッジの前に非同期クリア/セット/リセット信号が安定していなければならない最小時間。この時間が満たされていなければ、シーケンシャル・エレメントは正常な動作状態に入れないことがあります。リカバリー時間の解析と計算の方法はセットアップ時間と一致しています。このレポートは、コマンド report_timing -recovery によって生成されます。Gowin ソフトウェアは、デフォルトで 25 の最悪のスラックのタイミングパスを分析して報告します。レポートのヘッダーの詳細については、Setup Analysis Report を参照してください。

図 4-11 Recovery Analysis Report

Recovery Analysis Report

Report Command:report_timing -recovery -max_paths 25 -max_common_paths 1

Path1

Path Summary:

Slack	8.355
Data Arrival Time	4.629
Data Required Time	12.984
From	reg11_s0
To	reg12_s0
Launch Clk	sysyclk:[R]
Latch Clk	sysyclk:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.943	0.943	tINS	RR	2	IOL11[A]	clk_ibuf/O
3.236	2.293	tNET	RR	1	R2C9[0][A]	reg11_s0/CLK
3.786	0.550	tC2Q	RF	1	R2C9[0][A]	**reg11_s0/Q**
4.629	0.843	tNET	FF	1	R2C9[1][A]	**reg12_s0/CLEAR**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
10.000	10.000					active clock edge time
10.000	0.000					sysyclk
10.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
10.943	0.943	tINS	RR	2	IOL11[A]	clk_ibuf/O
13.236	2.293	tNET	RR	1	R2C9[1][A]	reg12_s0/CLK
13.036	-0.200	tUnc				
12.984	-0.052	tSu		1	R2C9[1][A]	reg12_s0

Path Statistics:

Clock Skew	0.000
Setup Relationship	10.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 0.843, 60.531%; tC2Q: 0.550, 39.469%
Required Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%

Removal Analysis Report

図 4-12 はリムーバル時間解析レポートです。リムーバル時間：シーケンシャル・エレメントの有効クロックエッジの後に非同期クリア/セット/リセット信号が安定していなければならない最小時間。この時間が満たされていなければ、シーケンシャル・エレメントは正常な動作状態に入れないことがあります。リムーバル時間の解析と計算の方法はホールド時間と一致しています。このレポートは、コマンド report_timing -removal によって生成されます。Gowin ソフトウェアは、デフォルトで 25 の最悪のスラックのタイミングパスを分析して報告します。レポートのヘッダーの詳細については、Hold Analysis Report を参照してください。

図 4-12 Removal Analysis Report

Removal Analysis Report

Report Command:report_timing -removal -max_paths 25 -max_common_paths 1

Path1

Path Summary:

Slack	1.003
Data Arrival Time	3.554
Data Required Time	2.551
From	reg11_s0
To	reg12_s0
Launch Clk	sysyclk:[R]
Latch Clk	sysyclk:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.811	0.811	tINS	RR	2	IOL11[A]	clk_ibuf/O
2.533	1.723	tNET	RR	1	R2C9[0][A]	reg11_s0/CLK
2.933	0.400	tC2Q	RR	1	R2C9[0][A]	**reg11_s0/Q**
3.554	0.621	tNET	RR	1	R2C9[1][A]	**reg12_s0/CLEAR**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.811	0.811	tINS	RR	2	IOL11[A]	clk_ibuf/O
2.533	1.723	tNET	RR	1	R2C9[1][A]	reg12_s0/CLK
2.533	0.000	tUnc				
2.551	0.018	tHld		1	R2C9[1][A]	reg12_s0

Path Statistics:

Clock Skew	0.000
Hold Relationship	0.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.811, 31.998%; route: 1.723, 68.002%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 0.621, 60.818%; tC2Q: 0.400, 39.182%
Required Clock Path Delay	cell: 0.811, 31.998%; route: 1.723, 68.002%

4.2.4 Minimum Pulse Width Report

最小パルス幅レポートです。High レベルの最小パルスと Low レベルの最小パルスを含む、タイミング解析に関係するパス上のすべてのシーケンシャル・エレメントの最小パルス幅を解析します。図 4-13 に示すとおりです。

- Actual Width：実際のパルス幅。この値は、解析対象で実際に維持されているパルス幅であり、つまり Early clock Path から Late clock Path を引いた値です。
- Required Width：必要な最小認識幅、つまりパルス信号が維持される最小時間。
- Slack：パルス幅のスラック。Actual Width から Required Width を引いた値です。計算式は $Slack = (T_{Early_clock_Path} - T_{Late_clock_Path}) - T_{\text{Required Width}}$。
- Type：パルスのタイプ。Low Pulse Width と High Pulse Width の 2 つのタイプがあります。

- Clock：静的タイミング解析用のクロック。
- Objects：現在解析されているシーケンシャル・エレメント。
- Late clock Path：クロックが最も遅く到着するパス。
- Early clock Path：クロックが最も早く到着するパス。

図 4-13 Minimum Pulse Width Report

MPW Summary:

Slack:	2.738
Actual Width:	4.238
Required Width:	1.500
Type:	Low Pulse Width
Clock:	sysclk1
Objects:	reg12_Z

Late clock Path:

AT	DELAY	TYPE	RF	NODE
5.000	0.000			active clock edge time
5.000	0.000			sysclk1
5.000	0.000	tCL	FF	clk1_ibuf/I
5.945	0.945	tINS	FF	clk1_ibuf/O
8.295	2.350	tNET	FF	reg12_Z/CLK

Early clock Path:

AT	DELAY	TYPE	RF	NODE
10.000	0.000			active clock edge time
10.000	0.000			sysclk1
10.000	0.000	tCL	RR	clk1_ibuf/I
10.811	0.811	tINS	RR	clk1_ibuf/O
12.533	1.723	tNET	RR	reg12_Z/CLK

4.2.5 High Fanout Nets Report

High Fanout Nets Report は、タイミング解析に関係するパス上の net のファンアウトを解析すると同時に、この net の最小 Slack と最大遅延時間を解析します。デフォルトでは 10 本解析されます。図 4-14 に示すとおりです(FANOUT 値の大きい順)。

- FANOUT：ファンアウト。
- NET NAME：net 名。
- WORST SLACK：net 上の最悪のスラック。net には複数のスラックが存在する可能性があります。
- MAX DELAY：net の最大遅延時間。

図 4-14 High Fanout Nets Report

High Fanout Nets Report:

Report Command:report_high_fanout_nets -max_nets 10

FANOUT	NET NAME	WORST SLACK	MAX DELAY
2	clk1_c	5.789	2.350
2	clk2_c	17.616	2.350
1	reg21_i	17.616	0.000
1	reg11	5.789	2.981
1	reg21	17.616	0.403

4.2.6 Route Congestions Report

図 4-15 は配線の密集レベルレポートです。

- GRID LOC：Grid の位置。
- ROUTE CONGESTIONS：Grid 上の配線の密集レベル。とえば、0.056 は、5.6%の密集レベルを示します。
- デフォルトでは、最悪の 10 個が報告され、ROUTE CONGESTIONS 値の大きい順にリストされます。

図 4-15 Route Congestions Report

Route Congestions Report:

Report Command:report_route_congestion -max_grids 10

GRID LOC	ROUTE CONGESTIONS
R5C9	0.056
R2C1	0.028
R3C1	0.028
R3C9	0.028
R1C1	0.014
R5C1	0.014

4.2.7 Timing Exceptions Report

次に、実際のケースを使用してタイミング例外レポートを解説します。

図 4-16 のケースに合わせた特定の SDC ファイルを図 4-17 に示します。

図 4-16 テストケース

```
1  module timing(
2  output dout,
3  input din, clk1, clk2
4  );
5
6  reg reg11, reg12;
7  reg reg21, reg22;
8
9
0
1  always @(posedge clk1)
2  begin
3      reg11 <= din;
4      reg12 <= reg11;
5  end
6
7  always @(posedge clk2)
8  begin
9      reg21 <= din;
0      reg22 <= ~reg21;
1  end
2
3  assign dout = reg22 & reg12;
4
5  endmodule
```

図 4-17 Timing Exceptions 制約

```
create_clock -name sysclk1 -period 10 -waveform {0 5} [get_ports {clk1}]
create_clock -name sysclk2 -period 10 -waveform {0 5} [get_ports {clk2}]
set_max_delay -from [get_clocks {sysclk1}] -to [get_clocks {sysclk1}] 5
set_max_delay -from [get_clocks {sysclk2}] -to [get_clocks {sysclk2}] 4
```

図 4-17 に示すように、タイミング例外制約文 set_max_delay は、sysclk1 と sysclk2 の影響を受けるタイミングパスの最大絶対遅延時間値を、それぞれ 5ns と 4ns に設定します。set_max_delay は setup 解析に影響し、影響を受けるパスはデフォルトでタイミング例外レポートに表示されます。デフォルトのレポートを図 4-18 に示します。

図 4-18 Timing Exceptions Report

Timing Exceptions Report:

Setup Analysis Report

Report Command:report_exceptions -setup -max_paths 5 -max_common_paths 1

Timing Path Constraint[1]: set_max_delay -from [get_clocks {sysclk1}] -to [get_clocks {sysclk1}] 5

Path1

Path Summary:

Slack	0.789
Data Arrival Time	6.767
Data Required Time	7.556
From	reg11_Z
To	reg12_Z
Launch Clk	sysclk1:[R]
Latch Clk	sysclk1:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysclk1
0.000	0.000	tCL	RR	1	IOL7[A]	clk1_ibuf/I
0.943	0.943	tINS	RR	2	IOL7[A]	clk1_ibuf/O
3.236	2.293	tNET	RR	1	IOL2[B]	reg11_Z/CLK
3.786	0.550	tC2Q	RF	1	IOL2[B]	**reg11_Z/Q**
6.767	2.981	tNET	FF	1	R5C9[1][A]	**reg12_Z/D**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
5.000	5.000					active clock edge time
5.000	0.000					sysclk1
5.000	0.000	tCL	RR	1	IOL7[A]	clk1_ibuf/I
5.943	0.943	tINS	RR	2	IOL7[A]	clk1_ibuf/O
8.236	2.293	tNET	RR	1	R5C9[1][A]	reg12_Z/CLK
8.036	-0.200	tUnc				
7.556	-0.480	tSu		1	R5C9[1][A]	reg12_Z

Path Statistics:

Clock Skew	0.000
Setup Relationship	5.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 2.981, 84.423%; tC2Q: 0.550, 15.577%
Required Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%

Timing Path Constraint[14]: set_max_delay -from [get_clocks {sysclk2}] -to [get_clocks {sysclk2}] 4

Path1

Path Summary:

Slack	1.616
Data Arrival Time	4.940
Data Required Time	6.556
From	reg21_Z
To	reg22_Z
Launch Clk	sysclk2:[R]
Latch Clk	sysclk2:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysclk2
0.000	0.000	tCL	RR	1	IOL5[A]	clk2_ibuf/I
0.943	0.943	tINS	RR	2	IOL5[A]	clk2_ibuf/O
3.236	2.293	tNET	RR	1	R5C9[0][B]	reg21_Z/CLK
3.786	0.550	tC2Q	RR	1	R5C9[0][B]	**reg21_Z/Q**
4.189	0.403	tNET	RR	1	R5C9[0][A]	reg21_i_cZ/I0
4.940	0.751	tINS	RF	1	R5C9[0][A]	reg21_i_cZ/F
4.940	0.000	tNET	FF	1	R5C9[0][A]	**reg22_Z/D**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
4.000	4.000					active clock edge time
4.000	0.000					sysclk2
4.000	0.000	tCL	RR	1	IOL5[A]	clk2_ibuf/I
4.943	0.943	tINS	RR	2	IOL5[A]	clk2_ibuf/O
7.236	2.293	tNET	RR	1	R5C9[0][A]	reg22_Z/CLK

タイミング例外レポートは、デフォルトでタイミング例外制約文の影響を受けるすべてのパスを報告します。Gowin ソフトウェアは、ユーザーが必要なレポートの一部のコンテンツを構成および表示し、不要なレポートパスをフィルタすることを可能にする report_exception 制約コマンド

を提供します。図 4-19 に示すように、赤いボックスの最初の行では、sysclk1 の影響を受けるパスの setup 解析が報告され、2 番目の行では、sysclk2 の影響を受けるパスの setup 解析が報告されません。

図 4-19 report_exception 文

```
create_clock -name sysclk1 -period 10 -waveform {0 5} [get_ports {clk1}]
create_clock -name sysclk2 -period 10 -waveform {0 5} [get_ports {clk2}]
set_max_delay -from [get_clocks {sysclk1}] -to [get_clocks {sysclk1}] 5
set_max_delay -from [get_clocks {sysclk2}] -to [get_clocks {sysclk2}] 4
report_exceptions -setup -from_clock [get_clocks {sysclk1}] -to_clock [get_clocks {sysclk1}] -max_paths 1 -max_common_paths 1
report_exceptions -setup -from_clock [get_clocks {sysclk2}] -to_clock [get_clocks {sysclk2}] -max_paths 0 -max_common_paths 0
```

図 4-19 の制約後のタイミング例外レポートを図 4-20 に示します。

図 4-20 report_exception レポート

Timing Exceptions Report:

Setup Analysis Report

Setup Analysis Report[1]:

Report Command:report_exceptions -setup -from_clock [get_clocks {sysclk1}] -to_clock [get_clocks {sysclk1}] -max_paths 1 -max_common_paths 1

Timing Path Constraint[1]: set_max_delay -from [get_clocks {sysclk1}] -to [get_clocks {sysclk1}] 5

Path1

Path Summary:

Slack	-0.654
Data Arrival Time	7.947
Data Required Time	7.293
From	reg11_ins23
To	reg12_ins20
Launch Clk	sysclk1:[R]
Latch Clk	sysclk1:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysclk1
0.000	0.000	tCL	RR	1	IOL15[A]	clk1_ibuf13/I
0.982	0.982	tINS	RR	2	IOL15[A]	clk1_ibuf13/O
2.893	1.911	tNET	RR	1	IOL2[B]	reg11_ins23/CLK
3.351	0.458	tC2Q	RF	1	IOL2[B]	**reg11_ins23/Q**
7.947	4.596	tNET	FF	1	R15C23[1][A]	**reg12_ins20/D**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
5.000	5.000					active clock edge time
5.000	0.000					sysclk1
5.000	0.000	tCL	RR	1	IOL15[A]	clk1_ibuf13/I
5.982	0.982	tINS	RR	2	IOL15[A]	clk1_ibuf13/O
7.893	1.911	tNET	RR	1	R15C23[1][A]	reg12_ins20/CLK
7.693	-0.200	tUnc				
7.293	-0.400	tSu		1	R15C23[1][A]	reg12_ins20

Path Statistics:

Clock Skew	0.000
Setup Relationship	5.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.982, 33.942%; route: 1.911, 66.058%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 4.596, 90.932%; tC2Q: 0.458, 9.068%
Required Clock Path Delay	cell: 0.982, 33.942%; route: 1.911, 66.058%

4.2.8 Timing Constraints Report

図 4-21 はタイミング制約レポートです。

- SDC Command Type：静的タイミング制約コマンドのタイプです。
- State：Invalid と Actived の 2 つの値があります。Actived はコマンドが有効であることを示し、Invalid はコマンドが無効であることを示します。

- Detail Command：SDC ファイル内の対応するタイミング制約文です。

図 4-21 Timing Constraints Report

Timing Constraints Report:

SDC Command Type	State	Detail Command
TC_CLOCK	Actived	create_clock -name main -period 18.182 -waveform {0 9.091} [get_ports {clk}]
TC_GENERATED_CLOCK	Actived	create_generated_clock -name main_gen -source [get_ports {clk}] -master_clock main -divide_by 5 -duty_cycle 40 -phase 22 -offset 50 [get_ports {in}]
TC_INPUT_DELAY	Actived	set_input_delay -clock main_gen 0.2 -clock_fall -add_delay -source_latency_included [get_ports {in}]
TC_CLOCK_LATENCY	Actived	set_clock_latency -source 1.2 [get_clocks {main}]
TC_CLOCK_UNCERTAINTY	Actived	set_clock_uncertainty 2.3 -setup -from [get_clocks {main}] -to [get_clocks {main}]
TC_FALSE_PATH	Actived	set_false_path -from [get_clocks {main_gen}] -to [get_clocks {main_gen}]
TC_MULTICYCLE	Actived	set_multicycle_path -from [get_clocks {main_gen}] -to [get_clocks {main_gen}] -setup -end 3
TC_MAX_DELAY	Actived	set_max_delay -from [get_clocks {main}] -to [get_clocks {main}] 1.11
TC_CLOCK_GROUP	Actived	set_clock_groups -exclusive -group [get_clocks {main}] -group [get_clocks {main_gen}]

5 構文仕様

ワイルドカード文字「?」と「*」の使用がサポートされています。「?」は 1 文字の一致を実現し、「*」は 0 文字以上の一致(階層間の一致をサポート)を実現します。また、複数の行に分割されるタイミング制約もサポートされています。

5.1 オブジェクトのコレクション

表 5-1 コレクションの説明

コレクション	説明
get_regs	フリップフロップオブジェクトのみを取得します
get_pins	ピンオブジェクトを取得します。
get_ports	ポートオブジェクトを取得します。
get_clocks	クロックオブジェクトを取得します。
get_nets	net オブジェクトを取得します。
all_clocks	すべてのクロックオブジェクトを取得します。
all_inputs	すべての入力ポートを取得します。
all_outputs	すべての出力ポートを取得します。

5.2 クロック制約

5.2.1 create_clock

クロックを作成します。create_clock は derive_clocks よりも優先度が高いです。

構文

create_clock

、

-name <*clock_name*>

-period <*period_value*>

[-waveform <*edge_list*>]

<*objects*>

[-add]

パラメータ

表 5-2 create_clock のパラメータ

-name <clock_name>	クロック名。この名前はクロックの一意の識別子です。 同じ名前のクロックを作成した場合、後で作成されたクロックによって前のクロックが上書きされます。
-period <period_value>	クロックサイクル(ns)。パラメータ値は正の実数である必要があります。
-waveform <edge_list>	クロックサイクル全体内のクロック波形の立ち上がりエッジ時刻と立ち下がりエッジ時刻を ns 単位で指定します。この 2 つの時間は次第に増える非負の数であり、それらの差は 1 クロックサイクル未満である必要があります。 このオプションを省略すると、立ち上がりエッジが 0 ns、立ち下がりエッジが<period_value>/2 ns のデフォルトの波形が選択されます。 例：Example 5.1
<objects>	オブジェクトを作成します。get_ports、get_nets、get_regs、get_pins、all_inputs、および all_outputs がサポートされます。
-add	同じ<*objects*>上に複数のクロックを作成したい場合、2 番目以降の create_clock 文で-add パラメータを使用する必要があります。そうしないと、2 番目以降の create_clock 文は無視されます。 例：Example 5.2

例

Example 5.1

//ワイルドカード「?」を使用してポート clk 上に user_clk という名前のクロックを作成します。クロックサイクルは 10 ns、立ち上がりエッジは 0 ns、立ち下がりエッジは 5 ns です。

create_clock -name user_clk -period 10 -waveform {0 5} [get_ports

、

{c?k}]

Example 5.2

//同じポート clk 上に 2 つのクロックを作成する場合は、2 番目の文に-add を追加する必要があります。

create_clock -name user_clk0 -period 10 -waveform {0 5} [get_ports {clk}]

create_clock -name user_clk1 -period 20 -waveform {0 10} [get_ports {clk}] -add

5.2.2 create_generated_clock

派生クロックを作成します。

構文

create_generated_clock

-name <*clock_name*>

-source <*clock_source*>

[-master_clock <*clock*>]

[-edges <*edge_list*>]

[-edge_shift <*shift_list*>]

[-divide_by <*factor*>]

[-multiply_by <*factor*>]

[-duty_cycle <*percent*>]

[-phase <*degrees*>]

[-offset <*time*>]

[-invert]

[-add]

<*objects*>

パラメータ

表 5-3 create_generated_clock のパラメータ

-name <clock_name>	派生クロック名。この名前は派生クロックの一意の識別子です。 同じ名前の派生クロックを作成した場合、後で作成された派生クロックによって前の派生クロックが上書きされます。 派生クロック名がマスタークロック名と同じ場合、派生クロックの作成は無視されます。
-source <clock_source>	派生クロックのソース。get_ports、get_nets、get_regs、get_pins がサポートされます。 ソースに複数のクロックがある場合は、-master_clock オプションを使用して特定のクロックを指定する必要があります。
-master_clock <clock>	派生クロックのマスタークロック。 例：Example 5.3
-edges <edge_list>	この 3 つの次第に増える正整数で構成されるパラメータリストはそれぞれ、派生クロックの最初の立ち上がりエッジ、最初の立ち下がりエッジ、および 2 つ目の立ち上がりエッジを表します。このリスト内の整数は、最初の立ち上がりエッジは 1、次の立ち下がりエッジは 2、次の立ち上がりエッジは 3 など、「立ち上がりエッジ、立ち下がりエッジ」が交互に続くルールに従って、ソースクロックから抽出されたエッジを表します。 -edges は-divide_by、-multiply_by、-duty_cycle、-phase、および-offset と併用することはできません。 例：Example 5.4
-edge_shift <shift_list>	3 つの実数で構成されるこのリストは、派生クロックの 3 つのエッジのシフトを表します。このオプションは-edges オプションと共に使用されます。オフセットによって、エッジが隣接するエッジを超えてはなりません。 -edge_shift は-divide_by、-multiply_by、-duty_cycle、-phase、および-offset と併用することはできません。 例：Example 5.4
-divide_by <factor>	周波数分割数。正の整数である必要があります。デフォルト値は 1 です。 例：Example 5.3、Example 5.6、Example 5.8

、

-multiply_by <factor>	周波数逓倍数。正の整数である必要があります。デフォルト値は 1 です。 例：Example 5.5、Example 5.8
-duty_cycle <percent>	デューティ サイクル。パラメータ値は(0, 100)で、デフォルトは 50%です。 例：Example 5.5
-phase <degrees>	位相。単位は度。デフォルトは 0 度です。正位相の場合、派生クロックはマスタークロックに対して右にシフトされます。負位相の場合、派生クロックはマスタークロックに対して左にシフトされます。 例：Example 5.8
-offset <time>	オフセット。単位は ns。デフォルトは 0 ns です。正の実数の場合、派生クロックはマスタークロックに対して指定されたオフセット分(ns)右にシフトされます。負の実数の場合、派生クロックはマスタークロックに対して指定されたオフセット分（ns）左にシフトされます。 例：Example 5.7
-invert	反転。 例：Example 5.7、Example 5.8
-add	同じオブジェクトに 2 つ以上の異なるクロックを作成する場合は、-add オプションを指定する必要があります。そうしないと、後で作成されたクロックは無視されます。 例：Example 5.8
<objects>	オブジェクト。get_ports、get_nets、get_regs、get_pins、all_inputs、および all_outputs がサポートされます。

例

Example 5.3

//ポート clk0 上に 2 つのマスタークロック user_clk0 と user_clk1 を作成します。ポート clk1 上に 2 分周の派生クロック user_clk2 を作成し、その派生クロックのマスタークロックとして user_clk1 を指定します。

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 20 [get_ports {clk0}] -add
```

、

create_generated_clock -name user_clk2 -source [get_ports {clk0}] -master_clock user_clk1 -divide_by 2 [get_ports {clk1}]

Example 5.4

//-edges オプションを使用して、各エッジが 0.2 ns シフトされた 2 分周クロックに相当する派生クロックを作成します。

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -edges {1 3 5} -edge_shift {0.2 0.2 0.2} [get_ports {clk1}]

Example 5.5

//3 逓倍のデューティサイクル 40%の派生クロックを作成します。

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -multiply_by 3 -duty_cycle 40 [get_ports {clk}]

Example 5.6

//2 分周の位相シフト 90 度の派生クロックを作成します

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -divide_by 2 -phase 90 [get_ports {clk1}]

Example 5.7

//マスタークロックに対して反転した右に 4 ns オフセットした派生クロックを作成します

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -offset 4 -invert [get_ports {clk1}]

Example 5.8

//ポート clk0 上に 2 分周、3 逓倍、反転出力である 2 番目の派生クロックを作成します。-add を追加する必要があります。

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -divide_by 2 -phase 90 [get_ports {clk1}]

create_generated_clock -name user_clk2 -source [get_ports {clk0}] -divide_by 2 -multiply_by 3 -invert -add [get_ports {clk1}]

、

5.2.3 set_clock_latency

クロック遅延時間を設定します。

構文

set_clock_latency

-source

[-rise | -fall]

[-late | -early]

<delay>

<objects>

[-clock *<clock_list>*]

パラメータ

表 5-4 set_clock_latency のパラメータ

-source	解析タイプをクロックソース遅延時間として指定します。
-rise | -fall	クロックの立ち上がりまたは立ち下がりエッジの遅延時間。 両方を省略した場合、デフォルトは-rise と-fall の両方に有効です。 -rise のみが設定されている場合、-fall は同じ値を使用します。-fall にも同じことが適用されます。 例：Example 5.9
-late | -early	クロックの最大(late)または最小(early)の遅延時間。 setup と recovery の解析では、データ到着パスには最大クロック遅延を使用し、データ所要パスには最小クロック遅延を使用します。 hold と removal の解析では、データ到着パスには最小クロック遅延を使用し、データ所要パスには最大クロック遅延を使用します。 -late のみが設定されている場合、-early は同じ値を使用します。-early にも同じことが適用されます。 -early 値が-late 値より大きい場合、-early は-late 値を使用します。 両方を省略した場合、デフォルトは-late と-early の両方に有効です。 例：Example 5.10

、

<delay>	実数、単位は ns。
<objects>	オブジェクト。get_clocks、get_ports、get_nets、get_regs、get_pins、および all_clocks がサポートされます。
-clock <clock_list>	クロック。-clock オプションが指定されていない場合、デフォルトは<objects>内のすべてのクロックに対して有効になります。 例：Example 5.11

例

Example 5.9

// user_clk0 と user_clk1 はそれぞれ送信クロックとサンプリングクロックです。user_clk0 の遅延時間を 0.22ns に設定し、user_clk1 の遅延時間を 0.33ns に設定します。0.22ns は Data Arrival Path、0.33ns は Data Required Path で確認できます。

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 10 [get_ports {clk1}]
set_clock_latency -source 0.22 -rise [get_clocks { user_clk0}]
set_clock_latency -source 0.33 -fall [get_clocks { user_clk1}]
```

Example 5.10

//クロック user_clk の最大遅延時間を 0.33ns、最小遅延時間を 0.11ns に設定します

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_clock_latency -source -late 0.33 [get_clocks {user_clk}]
set_clock_latency -source -early 0.11 [get_clocks {user_clk}]
```

Example 5.11

//ポート clk の user_clk1 の遅延時間を 0.5ns に設定します。これは、立ち上がりエッジ、立ち下がりエッジ、最大遅延、最小遅延に同時に有効です。

```
create_clock -name user_clk0 -period 10 [get_ports {clk}]
create_clock -name user_clk1 -period 20 [get_ports {clk}] -add
set_clock_latency -source 0.5 [get_ports {clk}] -clock [get_clocks {user_clk1}]
```

、

5.2.4 set_clock_uncertainty

クロックのばらつき値を設定します。

構文

set_clock_uncertainty

<uncertainty>

-setup | -hold

[-from *<from_clock>* | -rise_from *<rise_from_clock>* | -fall_from *<fall_from_clock>*]

[-to *<to_clock>* | -rise_to *<rise_to_clock>* | -fall_to *<fall_to_clock>*]

パラメータ

表 5-5 set_clock_uncertainty のパラメータ

<uncertainty>	実数、単位は ns。
-setup | -hold	-setup は、setup と recovery のばらつき解析を指定します。 -hold は、hold と removal のばらつき解析を指定します。 両方を省略した場合、setup、hold、recovery、removal にすべて有効になります。 例：Example 5.12、Example 5.13
-from *<from_clock>*	送信クロックのばらつき解析を実行します。get_clocks がサポートされます。 例：Example 5.12
-rise_from *<rise_from_clock>*	送信クロックの立ち上がりエッジのばらつき解析を実行します。get_clocks がサポートされます。 例：Example 5.13
-fall_from *<fall_from_clock>*	送信クロックの立ち下がりエッジのばらつき解析を実行します。get_clocks がサポートされます。 例：Example 5.14
-to *<from_clock>*	サンプリングクロックのばらつき解析を実行します。get_clocks がサポートされます。 同じタイミングパスに対して、to で指定した遅延値が -from、-rise_from、-fall_from で指定した遅延

、

	値と異なる場合は、-to で指定した値が使用されます。 例：Example 5.15
-rise_to *<rise_to_clock>*	サンプリングクロックの立ち上がりエッジのばらつき解析を実行します。get_clocks がサポートされます。 同じタイミングパスに対して、-rise_to で指定した遅延値が -from、-rise_from、-fall_from で指定した遅延値と異なる場合は、to で指定した値が使用されます。 例：Example 5.13
-fall_to *<fall_to_clock>*	サンプリングクロックの立ち下がりエッジのばらつき解析を実行します。get_clocks がサポートされます。 同じタイミングパスに対して、-fall_to で指定した遅延値が -from、-rise_from、-fall_from で指定した遅延値と異なる場合は、-to で指定した値が使用されます。 -from、-rise_from、-fall_from、-to、-rise_to、-fall_to オプションの少なくとも 1 つを指定する必要があります。 例：Example 5.14

例

Example 5.12

//setup と recovery の解析に対して、送信クロック user_clk のばらつき値を 0.2 ns に設定します。

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_clock_uncertainty 0.2 -setup -from [get_clocks {user_clk}]
```

Example 5.13

//hold と removal の解析に対して、送信クロック user_clk0 の立ち上がりエッジのばらつき値を 0.111 ns に設定し、サンプリングクロック user_clk1 の立ち上がりエッジのばらつき値を 0.222 ns に設定します。

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 10 [get_ports {clk1}]
```

、

```
set_clock_uncertainty 0.111 -hold -rise_from [get_clocks {user_clk0}]
set_clock_uncertainty 0.222 -hold -rise_to [get_clocks {user_clk1}]
```

Example 5.14

//hold と removal の解析に対して、送信クロック user_clk0 の立ち下がりエッジのばらつき値を 0.111 ns に設定し、サンプリングクロック user_clk1 の立ち下がりエッジのばらつき値を 0.222 ns に設定します。

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 20 [get_ports {clk1}]
set_clock_uncertainty -hold 0.111 -fall_from [get_clocks {user_clk0}]
set_clock_uncertainty -hold 0.222 -fall_to [get_clocks {user_clk1}]
```

Example 5.15

//setup と recovery の解析に対して、同じタイミングパスの送信クロック user_clk0 とサンプリングクロック user_clk1 のばらつき値をそれぞれ 0.2 ns と 0.3 ns に設定します。Gowin ソフトウェアは 0.3 ns で計算します。

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 10 [get_ports {clk1}]
set_clock_uncertainty -setup 0.2 -from [get_clocks {user_clk0}]
set_clock_uncertainty -setup 0.3 -to [get_clocks {user_clk1}]
```

、

5.2.5 set_clock_groups

クロックグループを設定します。

構文

set_clock_groups

-asynchronous | -exclusive

[-group <*clocks*>]

[-group <*clocks*>]

…

パラメータ

表 5-6 set_clock_groups のパラメータ

-asynchronous | -exclusive	-asynchronous または-exclusive：クロックグループ間の関係を非同期または相互排他として指定します。 例：Example 5.16、Example 5.17
-group <*clocks*>	クロックグループ。get_clocks を使用して取得します。

例

Example 5.16

//クロック clk1 と ck2 の間にタイミングパスがあります。それぞれ clk1 と clk2 のためにクロックグループと非同期関係を設定します。

create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk0}]

create_clock -name clk2 -period 10 -waveform {0 5} [get_ports {clk1}]

set_clock_groups -asynchronous -group [get_clocks {clk1}] -group [get_clocks {clk2}]

解析結果は下表に示すとおりです：

表 5-7 解析結果

送信クロック / サンプリングクロック	clk1	clk2
clk1	解析する	解析しない
clk2	解析しない	解析する

、

Example 5.17

//クロック clk1、ck2、clk3 の間に互いにタイミングパスがあります。clk1 と clk2 を 1 つのクロックグループとして設定し、clk3 を別のクロックグループとして設定します。

create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk0}]

create_clock -name clk2 -period 10 -waveform {0 5} [get_ports {clk1}]

create_clock -name clk3 -period 10 -waveform {0 5} [get_ports {clk2}]

set_clock_groups -exclusive -group [get_clocks {clk1 clk2}] -group [get_clocks {clk3}]

解析結果は下表に示すとおりです：

表 5-8 解析結果

送信クロック / サンプリングクロック	clk1	clk2	clk3
clk1	解析する	解析する	解析しない
clk2	解析する	解析する	解析しない
clk3	解析しない	解析しない	解析する

5.3 I/O 制約

データ入力とデータ出力の遅延時間値を設定します。

構文

set_input_delay / set_output_delay

-clock *<name>*

<delay_value>

[-clock_fall]

[-max | -min]

[-rise | -fall]

[-add_delay]

[-source_latency_included]

<objects>

、

パラメータ

表 5-9 I/O 制約のパラメータ

-clock *<name>*	クロックを指定します。このクロックは作成されたクロックでなければなりません。
<delay_value>	実数、単位はns。
-clock_fall	クロックの立ち下がりエッジを基準として指定します。 省略した場合、入力および出力遅延時間は、デフォルトで関連クロックの立ち上がりエッジを基準とします。 例：Example 5.18、Example 5.23
-max | -min	-maxは、最大入力および出力遅延時間を指定し、setupとrecoveryの解析に有効です。 -minは、最小入力および出力遅延時間を指定し、holdとremovalの解析に有効です。 -maxのみが設定されている場合、-minも同じ値になります。-minにも同じことが適用されます。 両方を省略した場合、最大および最小の入力および出力遅延時間は同じになります。 例：Example 5.19、Example 5.24
-rise | -fall	-riseまたは-fallは、入力ポートまたは出力ポートの立ち上がりエッジまたは立ち下がりエッジの遅延時間を指定します。 -riseのみが設定されている場合、-fallは同じ値になります。-fallにも同じことが適用されます。 両方を省略した場合、立ち上がりエッジと立ち下がりエッジの遅延時間は同じになります。 例：Example 5.20、Example 5.24
-add_delay	これにより、複数の制約を同時に有効にすることができます。ソフトウェアは設定された値に基づいてsetup(recovery)とhold(removal)解析を実行します。 例：Example 5.21、Example 5.26

、

-source_latency_included	入力および出力遅延時間にクロック ソースの遅延時間が含まれていることを指定します。このオプションを省略した場合、デフォルトでは、入力および出力遅延時間に関連クロックソース遅延時間は含まれません。 例：Example 5.22、Example 5.27
<objects>	入力および出力オブジェクト。get_ports、all_inputs/all_outputsがサポートされます。

注記：

set_input_delay はクロック入力 port をサポートしません。

例

Example 5.18

//関連クロック user_clk の立ち下がりエッジを基準として、入力ポート d の遅延時間値を 0.2ns に設定します。

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_input_delay -clock user_clk 0.2 -clock_fall [get_ports {d}]
```

Example 5.19

//関連クロック user_clk を基準として、入力ポート d の最大遅延時間値を 0.2 ns、最小遅延時間値を 0.1 ns に設定します。

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_input_delay -clock user_clk 0.2 -max [get_ports {d}]
set_input_delay -clock user_clk 0.1 -min [get_ports {d}]
```

Example 5.20

//関連クロック user_clk を基準として、入力ポート d の立ち上がりエッジ遅延時間値を 0.2 ns、立ち下がりエッジ遅延時間値を 0.1 ns に設定します。

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_input_delay -clock user_clk 0.2 -rise [get_ports {d}]
set_input_delay -clock user_clk 0.1 -fall [get_ports {d}]
```

Example 5.21

//関連クロック user_clk を基準として、入力ポート d の遅延時間を 0.2 ns と 0.4 ns に同時に設定します setup と recovery 解析の tIn の値は 0.4ns で、hold と removal 解析の tIn の値は 0.2ns です。

```
create_clock -name user_clk -period 10 [get_ports {clk}]
```

、

set_input_delay -clock user_clk 0.2 [get_ports {d}]

set_input_delay -clock user_clk 0.4 -add_delay [get_ports {d}]

Example 5.22

// 入力ポート d の遅延時間値を 0.2 ns に設定します。これに関連クロック user_clk のクロック遅延時間 0.15 ns が含まれています。

create_clock -name user_clk -period 10 [get_ports {clk}]

set_clock_latency -source 0.15 [get_ports {clk}] -clock [get_clocks {user_clk}]

set_input_delay -clock user_clk 0.2 -source_latency_included [get_ports {d}]

Example 5.23

//関連クロック user_clk の立ち下がりエッジを基準として、出力ポート q の遅延時間値を 0.2ns に設定します。

create_clock -name user_clk -period 10 [get_ports {clk}]

set_output_delay -clock user_clk 0.2 -clock_fall [get_ports {q}]

Example 5.24

//関連クロック user_clk を基準として、出力ポート q の最大遅延時間値を 0.2 ns、最小遅延時間値を 0.1 ns に設定します。

create_clock -name user_clk -period 10 [get_ports {clk}]

set_output_delay -clock user_clk 0.2 -max [get_ports {q}]

set_output_delay -clock user_clk 0.1 -min [get_ports {q}]

Example 5.25

//関連クロック user_clk を基準として、出力ポート q の立ち上がりエッジ遅延時間値を 0.2 ns、立ち下がりエッジ遅延時間値を 0.1 ns に設定します。

create_clock -name user_clk -period 10 [get_ports {clk}]

set_output_delay -clock user_clk 0.2 -rise [get_ports {q}]

set_output_delay -clock user_clk 0.1 -fall [get_ports {q}]

Example 5.26

//関連クロック user_clk を基準として、出力ポート q の遅延時間を 0.2 ns と 0.4 ns に同時に設定します。setup と recovery 解析の tOut の値は-0.2ns で、hold と removal 解析の tOut は-0.4ns です。

create_clock -name user_clk -period 10 [get_ports {clk}]

、

set_output_delay -clock user_clk 0.2 [get_ports {q}]

set_output_delay -clock user_clk 0.4 -add_delay [get_ports {q}]

Example 5.27

// 出力ポート q の遅延時間値を 0.2 ns に設定します。これに関連クロック user_clk のクロック遅延時間 0.15 ns が含まれています。

create_clock -name user_clk -period 10 [get_ports {clk}]

set_clock_latency -source 0.15 [get_ports {clk}] -clock [get_clocks {user_clk}]

set_output_delay -clock user_clk 0.2 -source_latency_included [get_ports {q}

5.4 例外制約

5.4.1 set_false_path

フォルスパスを設定します。設定されたタイミングパスはタイミング解析されません。

構文

set_false_path

[-from <*from_list*>]

[-through <*through_list*>]

[-to <*to_list*>]

[-setup | -hold]

パラメータ

表 5-10 set_false_path のパラメータ

-from <*from_list*>	始点。get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs、およびall_outputsがサポートされます。 例：Example 5.28、Example 5.29
-through <*through_list*>	通過点。get_netsおよびget_pinsがサポートされます。 例：Example 5.30
-to <*to_list*>	終点。get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs、およびall_outputsがサポートされます。 -from、-through、-toオプションの少なくとも1つを指定する必要があります。

、

	例：Example 5.28
-setup | -hold	-setupはsetupとrecovery解析に有効です。 -holdは、holdとremoval解析に有効です。 このオプションを省略した場合、setup、hold、recovery、removalの解析に対して有効になります。 例：Example 5.29、Example 5.30

例

Example 5.28

////送信クロックが user_clk0、サンプリングクロックが user_clk1 のすべてのタイミングパスをフォルスパスとして設定します。setup、hold、recovery、removal の解析に対して有効です。

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_clock -name user_clk1 -period 5 [get_ports {clk1}]

set_false_path -from [get_clocks {user_clk0}] -to [get_clocks {user_clk1}]

Example 5.29

//ポート d からのすべてのタイミングパスをフォルスパスとして設定します。setup と recovery の解析にのみ有効です。

set_false_path -from [get_ports {d}] -setup

Example 5.30

//FF reg0 の入力 D を通過するすべてのタイミングパスをフォルスパスとして設定します。hold と removal 解析にのみ有効です。

set_false_path -through [get_pins {reg0_s0/D}] -hold

5.4.2 set_max_delay / set_min_delay

最大/最小遅延時間値を設定します。

構文

set_max_delay / set_min_delay

[-from <*from_list*>]

[-through <*through_list*>]

、

[-to *<to_list>*]

<delay_value>

パラメータ

表 5-11 set_max_delay / set_min_delay のパラメータ

-from <from_list>	始点。get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs、および all_outputs がサポートされます。 例：Example 5.31
-through <through_list>	通過点。get_nets および get_pins がサポートされます。 例：Example 5.32
-to <to_list>	終点。get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs、および all_outputs がサポートされます。 -from、-through、-to オプションの少なくとも 1 つを指定する必要があります。 例：Example 5.31、Example 5.33
<delay_value>	実数、単位は ns。 最大遅延時間は、setup と recovery の解析に有効です。 最小遅延時間は、hold と removal の解析に有効です。

例

Example 5.31

//送信クロックが user_clk0、サンプリングクロックが user_clk1 のすべてのタイミングパスに対して最大遅延時間を 5ns、最小遅延時間 2 ns 設定します。5ns は setup と recovery 解析に有効で、2ns は hold と removal 解析に有効です。

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 5 [get_ports {clk1}]
set_max_delay -from [get_clocks {user_clk0}] 5 -to [get_clocks {user_clk1}]
set_min_delay -from [get_clocks {user_clk0}] 2 -to [get_clocks {user_clk1}]
```

Example 5.32

//FF reg0 の入力 D を通過するすべてのタイミングパスの最小遅延時間を 0.2 ns に設定します。

、

set_min_delay -through [get_pins {reg0_s0/D}] 0.2

Example 5.33

//FF reg0 が終点であるすべてのタイミングパスの最大遅延時間を 6 ns、最小遅延時間を 0.5 ns に設定します。

set_max_delay -to [get_regs {reg0_s0}] 6

set_min_delay -to [get_regs {reg0_s0}] 0.5

5.4.3 set_multicycle_path

マルチサイクルパスを設定します。

構文

set_multicycle_path

[-from <*from_list*>]

[-through <*through_list*>]

[-to <*to_list*>]

-setup | -hold

-start | -end

<*multiple_value*>

パラメータ

表 5-12 set_multicycle_path のパラメータ

-from <from_list>	始点。get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs、および all_outputs がサポートされます。 例：Example 5.34
-through <through_list>	通過点。get_nets および get_pins がサポートされます。 例：Example 5.35
-to <to_list>	終点。get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs、および all_outputs がサポートされます。 -from、-through、-to オプションの少なくとも 1 つを指定する必要があります。 例：Example 5.36

、

-setup | -hold	-setup は、setup と recovery 解析に有効です。 -hold は、hold と removal 解析に有効です。 例：Example 5.34、Example 5.36
-start | -end	-start または-end は、マルチサイクルの設定が送信クロックまたはサンプルクロックを基準とするかを指定します。 setup 解析では、-start は送信クロックを複数サイクル左にシフトし、-end はサンプリングクロックを複数サイクル右にシフトします。 hold 解析では、-start は送信クロックを複数サイクル右にシフトし、-end はサンプリングクロックを複数サイクル左にシフトします。 例：Example 5.34、Example 5.36
<multiple_value>	整数。

例

Example 5.34

//送信クロックが user_clk であるすべてのタイミングパスでは、サンプリングクロックの setup と recovery 解析に対して 3 サイクル解析を実行し、送信クロックの hold と removal 解析に対して 2 サイクル解析を実行します。

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_multicycle_path -from [get_clocks {user_clk}] -setup -end 3
set_multicycle_path -from [get_clocks {user_clk}] -hold -start 2
```

Example 5.35

// FF reg0 の出力 Q を通過するすべてのタイミングパスでは、サンプリングクロック setup と recovery 解析に対して 5 サイクル解析を実行します。

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_multicycle_path -through [get_pins {reg0_s0/Q}] 5 -setup
```

Example 5.36

//タイミングパスの送信クロックは user_clk0 で、サンプリングクロックは user_clk1 です。マイクルサイクルをそれぞれ 2 と 3 に設定し、setup と recovery 解析に有効です。 Gowin ソフトウェアは 2 でマルチサイクルを設定します。

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
、
```

```
create_clock -name user_clk1 -period 20 [get_ports {clk1}]
set_multicycle_path -from [get_clocks { user_clk0}]   -setup -end 2
set_multicycle_path -to   [get_clocks { user_clk1}]   -setup -end 3
```

5.5 レポート制約

5.5.1 report_timing

タイミングを報告します。

構文

report_timing

-setup | -hold | -recovery | -removal

[-from_clock <*clock_name*> | -rise_from_clock <*clock_name*> | -fall_from_clock <*clock_name*>]

[-to_clock <*clock_name*> | -rise_to <*clock_name* > | -fall_to <*clock_name*>]

[-from <*from_list*> | -rise_from <*rise_from_list*> | -fall_from <*fall_from_list*>]

[-through <*through_list*>]

[-to <*to_list*> | -rise_to <*rise_to_list*> | -fall_to <*fall_to_list*>]

[-max_paths <*value*>]

[-max_common_paths <*value*>]

[-max_logic_level <*value*>]

[-min_logic_level <*value*>]

[-mod_ins {mod_ins1 mod_ins2 ...}]

パラメータ

表 5-13 report_timing のパラメータ

-setup | -hold | -recovery | -removal	-setup、-hold、-recovery、-removal は、それぞれ setup、hold、recovery、removal 解析に有効です。 例：Example 5.37、Example 5.38、Example 5.39、Example 5.40
-from_clock <clock_name>	送信クロック。get_clocks がサポートされます。 例：Example 5.37

、

-rise_from_clock <clock_name>	送信クロック、立ち上がりエッジで有効。get_clocks がサポートされます。
-fall_from_clock <clock_name>	送信クロック、立ち下がりエッジで有効。get_clocks がサポートされます。
-to_clock <clock_name>	サンプリングクロック。get_clocks がサポートされます。 例： Example 5.37
-rise_to_clock <clock_name>	サンプリングクロック、立ち上がりエッジで有効。get_clocks がサポートされます。
-fall_to_clock <clock_name>	サンプリングクロック、立ち下がりエッジで有効。get_clocks がサポートされます。
-from <from_list>	始点。get_clocks、get_ports、get_regs、および get_pins がサポートされます。 Example 5.38
-rise_from <rise_from_list>	オブジェクトの送信クロック、立ち上がりエッジで有効。get_clocks がサポートされます。
-fall_from <fall_from_list>	オブジェクトの送信クロック、立ち下がりエッジで有効。get_clocks がサポートされます。
-through <through_list>	通過点。get_nets および get_pins がサポートされます。 Example 5.39
-to <to_list>	終点。get_clocks、get_ports、get_regs、および get_pins がサポートされます。 Example 5.38
-rise_to <rise_to_list>	オブジェクトのサンプリングクロック、立ち上がりエッジで有効。get_clocks と all_clocks がサポートされます。
-fall_to <fall_to_list>	オブジェクトのサンプリングクロック、立ち下がりエッジで有効。get_clocks と all_clocks がサポートされます。
-max_paths <value>	出力するタイミングパスの最大数(正の整数)。 このオプションを省略した場合、デフォルトは 25 になります。 Example 5.38

、

-max_common_paths <value>	出力する共通パスの最大数(正の整数)。デフォルトは 1 です。 同じ終点とみなされるには、同じクロックと同じオブジェクトという条件を同時に満たす必要があります。 Example 5.38
-max_logic_level <value>	ロジックレベル数が設定値（正の整数）以下のタイミングパスを出力します。 Example 5.37
-min_logic_level <value>	ロジックレベル数が設定値（正の整数）以上のタイミングパスを出力します。 Example 5.39
-mod_ins {mod_ins1 mod_ins2 …}	デザイン内のインスタンス化されたサブモジュールに関連付けられたタイミングパスを出力します。 Example 5.40

例

Example 5.37

//setup 部分で、送信クロックとサンプリングクロックが user_clk でロジックレベル数が 4 未満のタイミングパスを出力するよう制約します。

create_clock -name user_clk -period 10 [get_ports {clk}]

report_timing -setup -from_clock [get_clocks {user_clk}] -to_clock [get_clocks {user_clk}] -max_logic_level 4

Example 5.38

//hold 部分で、FF reg0 と reg1 間のタイミングパスを出力します。最大 50 本のパスを出力し、最大 5 本の共通パスを出力します。

report_timing -hold -from [get_regs {reg0_s0}] -to [get_regs {reg1_s0}] -max_paths 50 -max_common_paths 5

Example 5.39

//recovery 部分で、FF reg0 の入力 D を通過する、ロジックレベル数が 3 より大きいタイミングパスを出力します。

report_timing -recovery -through [get_pins {reg0_s0/D}] -min_logic_level 3

Example 5.40

//removal 部分で、デザイン内のインスタンス化されたモジュールに関連付けられたタイミングパスを出力します。

、

report_timing -removal -mod_ins {sub_inst}

5.5.2 report_high_fanout_nets

高ファンアウトの net を報告します

構文

report_high_fanout_nets

-max_nets <*max_nets_value*>

[-min_fanout <*min_fanout_value*>]

[-max_fanout <*max_fanout_value*>]

[-clock_regions]

[-slr]

[-ascending]

パラメータ

表 5-14 report_high_fanout_nets のパラメータ

-max_nets <max_nets_value>	net の最大数。正の整数である必要があります。デフォルト値は 10 です。
-min_fanout <min_fanout_value>	ファンアウト数が設定値より大きい net を出力します。正の整数である必要があります。-max_fanout 値以下である必要があります。 例：Example 5.41
-max_fanout <max_fanout_value>	ファンアウト数が設定値より小さい net を出力します。正の整数である必要があります。-min_fanout 値以上である必要があります。 例：Example 5.42
-clock_regions	クロックポートに接続されている net を出力します。 例：Example 5.42
-slr	LSR ポートに接続されている net を出力します。LSR はリセット、セット、クリアを意味します。 例：Example 5.43
-ascending	net のファンアウト値の昇順で出力します。 このオプションを省略した場合、デフォルトでは net のファンアウト値の降順で出力されます。 例：Example 5.43

、

例

Example 5.41

//ファンアウト数が 2 以上の net を出力します。最大 25 本の net を出力します

report_high_fanout_nets -max_nets 25 -min_fanout 2

Example 5.42

//clk に関連する、ファンアウト数が 3 以下の net を出力します。

report_high_fanout_nets -max_fanout 3 -clock_regions

Example 5.43

//セットとリセットに関連する net を、ファンアウト数の昇順で出力します。

report_high_fanout_nets -slr -ascending

5.5.3 report_route_congestion

配線の密集レベルを報告します。

構文

report_route_congestion

-max_grids <*max_grids_value*>

[-min_route_congestion <*min_route_congestion_value*>]

[-max_route_congestion <*max_route_congestion_value*>]

[-LOC <*grid_position*>]

パラメータ

表 5-15 report_route_congestion のパラメータ

-max_grids <max_grids_value>	出力する grid の最大数。正の整数である必要があります。デフォルト値は 10 です。 例：Example 5.44
-min_route_congestion <min_ route_congestion _value>	配線の密集レベルが設定値以上の grid を出力します。パラメータ値は [0,1]の実数であり、-max_route_congestion のパラメータ値以下である必要があります。 例：Example 5.45

、

-max_route_congestion <max_ route_congestion _value>	配線の密集レベルが設定値以下の grid を出力します。パラメータ値は [0,1]の実数であり、-min_route_congestion のパラメータ値以上である必要があります。 例：Example 5.45
-LOC <grid_position>	制約位置範囲内の grid を出力します。 例：Example 5.44

例

Example 5.44

//3 行目から 5 行目、4 列目から 6 列目の範囲の grid を出力します。最大 30 個を出力します。

report_route_congestion -max_grids 30 -LOC R[3:5]C[4:6]

Example 5.45

// 0.5 から 1 の配線の密集レベルを持つ grid を出力します。最大 10 個を出力します。

report_route_congestion -max_grids 30 -min_route_congestion 0.5 -max_route_congestion 1

5.5.4 report_min_pulse_width

最小パルス幅を報告します。

構文

report_min_pulse_width

-nworst <nworst_value>

[-min_pulse_width <*min_pulse_width_value*>]

[-max_pulse_width <*max_pulse_width_value*>]

[-detail]

<*objects*>

パラメータ

表 5-16 report_min_pulse_width のパラメータ

-nworst <nworst_value>	出力する FF の最悪 MPW の最大数。正の整数である必要があります。デフォルト値は 10 です。 例：Example 5.46、Example 5.47、Example 5.48

、

-min_pulse_width <min_pulse_width_value>	実際の MPW が設定値以上の FF を出力します。 この設定値は正の実数であり、-max_pulse_width のパラメータ値以下である必要があります。 例：Example 5.46、Example 5.47
-max_pulse_width <max_pulse_width_value>	実際の MPW が設定値以上の FF を出力します。 この設定値は正の実数であり、-min_pulse_width のパラメータ値以上である必要があります。 例：Example 5.47
-detail	要約テーブルに加えて、クロックパス解析を含む、FF の MPW に関する詳細情報を出力します。 このオプションを省略した場合、要約テーブルのみが印刷されます。 例：Example 5.46
<objects>	特定の FF の MPW のみを出力します。get_regs がサポートされます。 例：Example 5.48

例

Example 5.46

//実際の MPW が 2.5 以上の FF を出力します。最大 30 個を出力します。クロックパス解析を出力します。

report_min_pulse_width -nworst 30 -min_pulse_width 2.5 -detail

Example 5.47

//実際の MPW が 2～4 の FF を出力します。最大 15 個を出力します。

report_min_pulse_width -nworst 15 -min_pulse_width 2 -max_pulse_width 4

Example 5.48

//reg0 と reg1 の MPW のみを出力します。最大 10 個を出力します。

report_min_pulse_width -nworst 10 [get_regs {reg0_s0 reg1_s0}]

5.5.5 report_max_frequency

サブモジュールの最大動作周波数を報告します。

タイミングレポートにデザイン内のサブモジュール関連の同期クロックの最大動作周波数を出力します。この最大動作周波数は、同期タイミン

、

グパスの終点がサブモジュール内にある最悪のタイミングパスに基づいて計算されます。

構文

report_max_frequency

-mod_ins {mod_ins0 mod_ins1 ...}

パラメータ

表 5-17 report_max_frequency のパラメータ

-mod_ins {mod_ins0 mod_ins1 …}	デザイン内のサブモジュール関連のクロックの最大動作周波数を出力します。 例：Example 5.49

例

Example 5.49

//デザイン内の mod_inst0 関連の同期クロックの最大動作周波数を出力します

report_max_frequency -mod_ins {mod_inst0}

5.5.6 report_exceptions

例外制約を報告します。

構文の例については、report_timing を参照してください。

構文

report_exceptions

-setup | -hold | -recovery | -removal

[-from_clock <*clock_name*> | -rise_from_clock <*clock_name*> | -fall_from_clock <*name*>]

[-to_clock <*clock_name*> | -rise_to <*clock_name*> | -fall_to <*clock_name*>]

[-from <*from_list*> | -rise_from <*rise_from_list*> | -fall_from <*fall_from_list*>]

[-through <*through_list*>]

[-to <*to_list*> | -rise_to <*rise_to_list*> | -fall_to <*fall_to_list*>]

[-max_paths <*value*>]

、

[-max_common_paths <*value*>]

[-max_logic_level <*value*>]

[-min_logic_level <*value*>]

パラメータ

表 5-18 report_exceptions のパラメータ

-setup | -hold | -recovery | -removal	-setup、-hold、-recovery、-removal は、それぞれ setup、hold、recovery、removal 解析に有効です。
-from_clock <name>	送信クロック。get_clocks がサポートされます。
-rise_from_clock <name>	送信クロック、立ち上がりエッジで有効。get_clocks がサポートされます。
-fall_from_clock <name>	送信クロック、立ち下がりエッジで有効。get_clocks がサポートされます。
-to_clock <name>	サンプリングクロック。get_clocks がサポートされます。
-rise_to_clock <name>	サンプリングクロック、立ち上がりエッジで有効。get_clocks がサポートされます。
-fall_to_clock <name>	サンプリングクロック、立ち下がりエッジで有効。get_clocks がサポートされます。
-from <from_list>	始点。get_clocks、get_ports、get_regs、および get_pins がサポートされます。
-rise_from <rise_from_list>	オブジェクトの送信クロック、立ち上がりエッジで有効。get_clocks がサポートされます。
-fall_from <fall_from_list>	オブジェクトの送信クロック、立ち下がりエッジで有効。get_clocks がサポートされます。
-through <through_list>	通過点。get_nets および get_pins がサポートされます。

、

-to <to_list>	終点。get_clocks、get_ports、get_regs、および get_pins がサポートされます。
-rise_to <rise_to_list>	オブジェクトのサンプリングクロック、立ち上がりエッジで有効。get_clocks と all_clocks がサポートされます。
-fall_to <fall_to_list>	オブジェクトのサンプリングクロック、立ち下がりエッジで有効。get_clocks と all_clocks がサポートされます。
-max_paths <value>	出力するタイミングパスの最大数(正の整数)。デフォルトは 25 です。
-max_common_paths <value>	出力する共通パスの最大数(正の整数)。デフォルトは 1 です。 同じ終点とみなされるには、同じクロックと同じオブジェクトという条件を同時に満たす必要があります。
-max_logic_level <value>	ロジックレベル数が設定値（正の整数）より小さいタイミングパスを出力します。
-min_logic_level <value>	ロジックレベル数が設定値（正の整数）より大きいタイミングパスを出力します。

5.6 動作条件の制約

動作条件を設定します。

構文

set_operating_conditions

-grade <*c* | *i* | *a*>

-model <*slow* | *fast*>

-speed <*speed_value*>

[-setup]

[-hold]

パラメータ

表 5-19 動作条件制約パラメータ

-grade <c | i | a>	デバイスの温度グレードを指定します。このうち、c はコマーシャル(commercial)、i はインダストリアル(industrial)、a はオートモーティブ(automotive)を表します。 例：Example 5.50、Example 5.51、Example 5.52

、

-model <slow | fast>	タイミング解析に使用する遅延モデルを指定します。 例：Example 5.50、Example 5.51、Example 5.52
-speed <speed_value>	デバイスのスピードグレードを指定します。パラメータ値は正の整数であり、デバイスでサポートされているスピードグレードと一致する必要があります。
-setup	setup と recovery 解析に、設定された遅延モデルを適用します。このオプションを省略した場合、デフォルトで Slow Model(低速遅延時間モデル)が使用されます。 例：Example 5.50、Example 5.52
-hold	hold と removal 解析に、設定された遅延モデルを適用します。このオプションを省略した場合、デフォルトで Fast Model(高速遅延時間モデル)が使用されます。 例：Example 5.51、Example 5.52

例

Example 5.50

//デバイスの温度グレードをコマーシャル、スピードグレードを 7 に設定します。setup と recovery 解析に高速遅延時間モデルを使用します。

```
set_operating_conditions -grade c -model fast -speed 7 -setup
```

Example 5.51

//デバイスの温度グレードをインダストリアル、スピードグレードを 6 に設定します。hold と removal 解析に低速遅延時間モデルを使用します。

```
set_operating_conditions -grade i -model slow -speed 6 -hold
```

Example 5.52

//デバイスの温度グレードをオートモーティブ、スピードグレードを 6 に設定します。setup、hold、recovery、removal 解析に高速遅延時間モデルを使用します。

```
set_operating_conditions -grade a -model fast -speed 6 -setup -hold
```

5.7 その他の制約

5.7.1 derive_clocks

グローバルのクロック周波数を指定します。derive_clocks は、Configuration->Global->Frequency(Mhz)で設定された周波数よりも優先度

、

が高いです。

構文

derive_clocks

-freq <*freq_value*>

パラメータ

表 5-20 derive_clocks のパラメータ

-freq <*freq_value*>	グローバルのクロック周波数をMHz単位で指定します。パラメータ値は (0,1200]の実数である必要があります。デフォルト値は100です。 例：Example 5.53

例

Example 5.53

//グローバルのクロック周波数を 90MHz に指定します。

derive_clocks -freq 90

、

GOWIN高云

智慧逻辑 定制未来

、