

この資料は英語版を翻訳したもので、内容に相違が生じる場合には原文を優先します。こちらの日本語版は参考用としてご利用ください。設計の際には、最新の英語版で内容をご確認ください。

QI153018-8.0.0

はじめに

Quartus® II TimeQuest タイミング・アナライザは、ASIC スタイルの強力なタイミング解析ツールで、業界標準の制約、解析、およびレポート手法により、デザイン内のすべてのロジックのタイミング性能を検証します。Quartus II TimeQuest タイミング・アナライザの GUI またはコマンドライン・インタフェースを使用して、デザインのすべてのタイミング・パスを制約および解析し、結果をレポートします。

Quartus II TimeQuest タイミング・アナライザを実行する前に、クロック特性、タイミング例外、信号遷移の到達および所要時間を規定する初期タイミング制約を指定する必要があります。GUI またはコマンドライン・インタフェースを使用して、Synopsys Design Constraints (SDC) ファイル・フォーマットでタイミング制約を指定することができます。Quartus II フィッタは、ロジックの配置を最適化して制約条件を満たします。

Quartus II TimeQuest タイミング・アナライザは、タイミング解析時にデザインのタイミング・パスの解析、各パスでの伝播遅延の計算、タイミング制約違反のチェックを実行し、タイミング結果をスラックとして **Report** ペインおよび **Console** ペインでレポートします。Quartus II TimeQuest タイミング・アナライザがタイミング違反をレポートした場合は、レポートをカスタマイズして特定のパスに関する正確なタイミング情報を表示し、それらのパスを制約して違反を修正することができます。デザインにタイミング違反がない場合、ロジックはターゲット・デバイスで意図したとおり動作します。

Quartus II TimeQuest タイミング・アナライザは、アルテラ FPGA および HardCopy® ASIC に対するサイン・オフ・ツールとして使用できる完全なスタティック・タイミング解析ツールです。

この章は、以下の項で構成されています。

- 「Quartus II TimeQuest タイミング・アナライザの使用法」
- 7-3 ページの「Quartus II TimeQuest タイミング・アナライザのガイドラインに準拠したコンパイル・フロー」
- 7-7 ページの「タイミング解析の概要」
- 7-27 ページの「Quartus II TimeQuest タイミング・アナライザのフローに関するガイドライン」
- 7-29 ページの「コレクション」
- 7-31 ページの「SDC 制約ファイル」

- 7-33 ページの「クロックの仕様」
- 7-53 ページの「I/O 規格」
- 7-57 ページの「タイミング例外」
- 7-65 ページの「制約と例外の削除」
- 7-66 ページの「タイミング・レポート」
- 7-92 ページの「タイミング解析機能」
- 7-100 ページの「TimeQuest タイミング・アナライザ GUI」
- 7-113 ページの「まとめ」



TimeQuest タイミング・アナライザおよび SOPC Builder について詳しくは、「Quartus II ハンドブック Volume 4」の「SOPC Builder」の章を参照してください。

Quartus II TimeQuest タイミング・アナライザの使用法

Quartus II TimeQuest タイミング・アナライザは、FPGA の最も基本的なデザインから最も高度なデザインまでのニーズを満たします。

この項では、適切なデザインの制約、完全な配置配線、デザインに関するレポート作成に必要なステップなど、Quartus II TimeQuest タイミング・アナライザの概要を述べます。

Quartus II TimeQuest タイミング・アナライザの設定

Quartus II ソフトウェア v7.2 以降は、Quartus II TimeQuest タイミング・アナライザと Quartus II クラシック・タイミング・アナライザの2つのネイティブ・タイミング解析ツールを備えています。Quartus II TimeQuest タイミング・アナライザをデフォルトのタイミング解析ツールとして指定すると、Quartus II TimeQuest タイミング・アナライザはフィッタをガイドし、コンパイル後のタイミング結果を解析します。

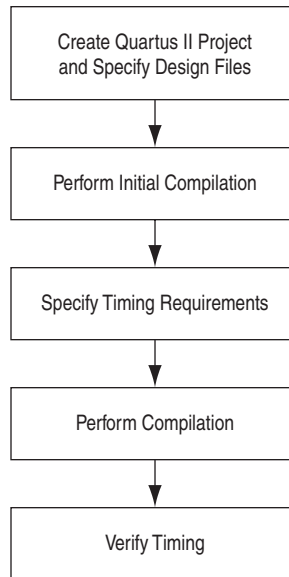
Quartus II TimeQuest タイミング・アナライザをデフォルトのタイミング・アナライザとして指定するには、Assignments メニューの **Settings** をクリックします。**Settings** ダイアログ・ボックスの **Category** リストで、**Timing Analysis Settings** を選択し、**Use TimeQuest Timing Analyzer during compilation** をオンにします。

Quartus II ツールバーに TimeQuest のアイコンを追加するには、Tools メニューの **Customize** をクリックします。**Customize** ダイアログ・ボックスで、**Toolbars** タブをクリックし、**Processing** をオンにし、**Close** をクリックします。

Quartus II TimeQuest タイミング・ アナライザの ガイドライン に準拠した コンパイル・ フロー

Quartus II TimeQuest タイミング・アナライザをデフォルトのタイミング・アナライザとしてイネーブルすると、制約検証からタイミング検証までのあらゆる処理が、Quartus II TimeQuest タイミング・アナライザによって実行されます。図 7-1 に、Quartus II TimeQuest タイミング・アナライザの利点を最大限に高め、利用するための推奨デザイン・フロー・ステップを示します。各ステップの詳しい説明は、この図の後にあります。

図 7-1. Quartus II TimeQuest タイミング・アナライザのデザイン・フロー



- **Quartus II** プロジェクトを作成し、デザイン・ファイルを指定する — デザイン・ファイルをコンパイルするには、まずプロジェクトを作成します。このステップでは、ターゲット FPGA を指定し、デザイン・サイクルで使用する EDA ツール、およびすべてのデザイン・ファイルを指定します。

また、デザインの最適化のために既存のデザイン・ファイルを修正したり、デザイン・ファイルを追加することもできます。例えば、プロジェクトに HDL ファイルや回路図を追加できます。

- 初期コンパイルを実行する—デザインのタイミング制約を指定する前に、初期デザイン・データベースを作成します。Analysis and Synthesis を実行して、マップ後データベースを作成するか、フル・コンパイルを実行して、フィッティング後データベースを作成します。

初期コンパイル用のマップ後データベースは、フィッティング後データベースよりも素早く作成されます。初期データベースにはマップ後データベースで十分です。

フィッティング後データベースの作成が推奨されるのは、プロジェクト用の SDC ファイルをすでに作成して指定済みである場合に限られます。初期コンパイルにはマップ後データベースで十分です。

- タイミング要求を指定する — フィットはデザインを配置配線する際、タイミング要求によってガイドされます。

SDC ファイルにタイミング制約と例外をすべて入力する必要があります。このファイルは、プロジェクトの一部として含める必要があります。このファイルをプロジェクトに追加するには、Project メニューの **Add/Remove Files in Project** をクリックし、Files ダイアログ・ボックスで SDC ファイルを追加します。

- コンパイルを実行する — デザインを合成し、配置し、ターゲット FPGA に配線します。

コンパイルが完了すると、TimeQuest タイミング・アナライザにより、デザインに定義されているすべてのクロックについて、クロック・セットアップとクロック・ホールドのサマリ、リカバリ、およびリムーバル・レポートが生成されます。

- タイミングを検証する — Quartus II TimeQuest タイミング・アナライザで、デザインのタイミングを検証します。7-27 ページの「[Quartus II TimeQuest タイミング・アナライザのフローに関するガイドライン](#)」を参照してください。

Quartus II TimeQuest タイミング・ アナライザ の実行


Quartus II TimeQuest タイミング・アナライザは、以下のいずれかのモードで実行できます。

- Quartus II ソフトウェアから直接
- スタンドアロン・モード
- コマンドライン・モード

この項では、これらの各モードと Quartus II TimeQuest タイミング・アナライザの動作について説明します。

Quartus II ソフトウェアから直接

Quartus II TimeQuest タイミング・アナライザを Quartus II ソフトウェアから実行するには、Tools メニューの **TimeQuest Timing Analyzer** をクリックします。Quartus II TimeQuest タイミング・アナライザは、現在のプロジェクトで使用するデータベースの作成後に使用可能になります。データベースは、マップ後データベースまたはフィッティング後データベースのどちらでも使用できます。マップ後データベースを作成するには、**Analysis and Synthesis** を実行し、フィッティング後データベースを作成するにはフル・コンパイルを実行します。

 データベースを作成すると、そのデータベースに基づいてタイミング・ネットリストを作成できます。マップ後データベースを作成すると、Quartus II TimeQuest タイミング・アナライザでフィッティング後タイミング・ネットリストは作成できません。

TimeQuest タイミング・アナライザを Quartus II ソフトウェアから直接起動すると、デフォルトで現在のプロジェクトが開きます。

スタンドアロン・モード

Quartus II TimeQuest タイミング・アナライザをスタンドアロン・モードで実行するには、コマンド・プロンプトで次のコマンドを入力します。

```
quartus_staw -J
```

スタンドアロン・モードでは、マップ後データベースまたはフィッティング後データベースのいずれかを含むプロジェクトに対して、スタティック解析を実行できます。プロジェクトを開くには、**Tasks** ペインの **Open Project** をダブルクリックします。

コマンドライン・モード

コマンドライン・モードを使用すると、スクリプト化されたデザイン・フローと簡単に統合できます。コマンドライン・モードでは、Quartus II TimeQuest タイミング・アナライザのユーザー・インタフェースは使用できませんが、スタティック・タイミング解析フローの各ステップを自動化できます。表 7-1 に、コマンドライン・モードで使用可能なオプションの概要を示します。

表 7-1. コマンドライン・オプションの概要	
コマンドライン・オプション	説明
-h --help	quartus_sta に関するヘルプ情報を表示します。
-t <script file> --script=<script file>	<script file> を参照します。
-s --shell	シェル・モードに入ります。
--tcl_eval <tcl command>	Tcl コマンド <tcl command> を評価します。
--do_report_timing	次のコマンドを実行します。 report_timing -npaths 1 -to_clock \$clock。デザイン内のすべてのクロックが対象です。
--force_dat	最近コンパイルされたデザインからコンパイラ・データベースまでの新しい遅延が、遅延アノテータによって強制的にアノテーションされます。
--lower_priority	quartus_sta プロセスを処理する優先度を低くします。
--post_map	マップ後データベースの結果を使用します。
--qsf2sdc	Quartus II 設定ファイル (.qsf) 形式から Synopsys Design Constraints ファイル形式にアサインメントを変換します。
--sdc=<SDC file>	読み出す SDC ファイルを指定します。
--fast_model	高速コーナー遅延モデルを使用します。
--report_script=<script>	カスタム・レポート・スクリプトを呼び出すよう指定します。
--speed=<value>	デバイスのスピード・グレードをタイミング解析で使用するように指定します。
--tq2hc	一時ファイルを生成し、Quartus II TimeQuest タイミング・アナライザの SDC ファイルを、HardCopy Design Center (HCDC) で使用できる PrimeTime SDC ファイルに変換します。
--tq2pt	一時ファイルを生成し、Quartus II TimeQuest タイミング・アナライザの SDC ファイルを PrimeTime SDC ファイルに変換します。
-f <argument file>	追加コマンドライン引数を含むファイルを指定します。
-c <revision name> --rev=<revision_name>	使用するリビジョンとその関連する Quartus II 設定ファイル (.qsf) を指定します。
--multicorner	低速および高速コーナーの両方について、すべてのスラック・サマリ・レポートを生成するように指定します。

Quartus II TimeQuest タイミング・アナライザをコマンドライン・モードで実行するには、コマンド・プロンプトで次のコマンドを入力します。

```
quartus_sta <options> ↵
```

タイミング解析の概要

この項では、Quartus II TimeQuest タイミング・アナライザのコンセプトの概要を説明します。ここで述べるコンセプトを理解すると、Quartus II TimeQuest タイミング・アナライザで提供される強力なタイミング解析機能を利用できるようになります。

Quartus II TimeQuest タイミング・アナライザは、図 7-2 に示すフローに従ってデザインを解析します。表 7-2 に、各ステップで最も一般的に使用されるコマンドを示します。

図 7-2. Quartus II TimeQuest タイミング・アナライザのフロー

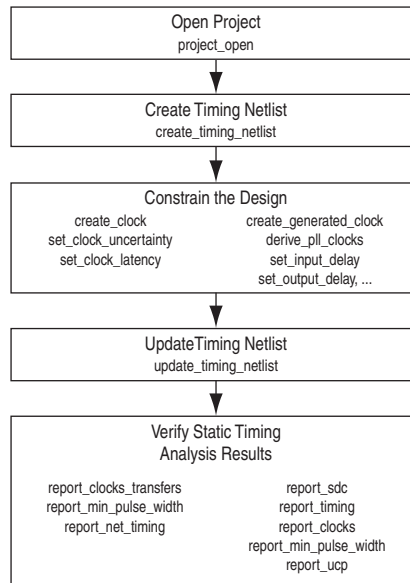


表 7-2 に、Quartus II TimeQuest タイミング・アナライザ用語を示します。

用語	定義
Nodes	最も基本的なタイミング・ネットリスト単位。ポート、ピン、およびレジスタを表すのに使用します。
Keepers	ポートまたはレジスタ。(1)
Cells	ルック・アップ・テーブル (LUT)、レジスタ、デジタル信号処理 (DSP) ブロック、TriMatrix メモリ、IOE など。(2)
Pins	セルの入力または出力。
Nets	ピン間の接続。
Ports	トップレベル・モジュールの入力または出力 (デバイス・ピンなど)。
Clocks	デザイン外の抽象オブジェクト。

表 7-2 の注：

- (1) ピンはキーパーを間接的に参照できます。例えば、制約の `-from` フィールドの値が専用メモリへのクロック・ピンの場合があります。この場合、クロック・ピンはレジスタの集合を参照します。
- (2) Stratix® デバイスとその他の初期デバイス・ファミリの場合、LUT とレジスタは、ロジック・エレメント (LE) に含まれており、これらのデバイス・ファミリのセルとして機能します。

Quartus II TimeQuest タイミング・アナライザで、デザインのタイミング解析を実行するには、タイミング・ネットリストが必要です。例えば、[図 7-3](#) に示すデザインの場合、Quartus II TimeQuest タイミング・アナライザは、[図 7-4](#) に示すのと同等のネットリストを生成します。

図 7-3. デザイン例

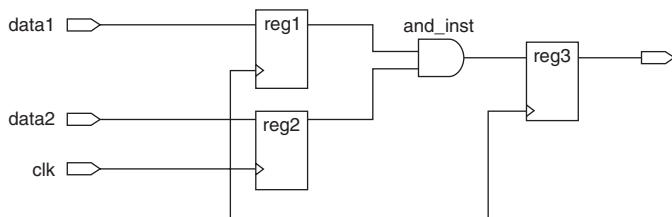


図 7-4. Quartus II TimeQuest タイミング・アナライザのタイミング・ネットリスト

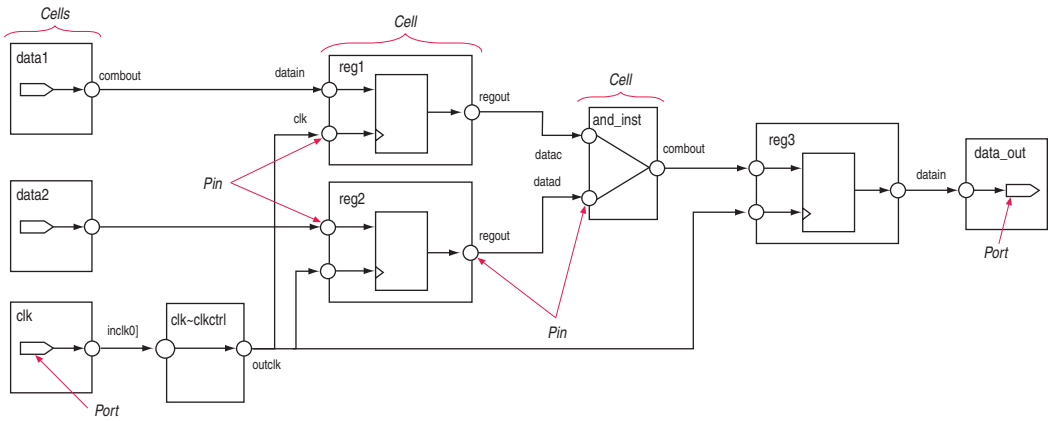


図 7-4 に、各種セル、ピン、ネット、およびポートを示します。次のサンプル・セル名が使用されます。

- reg1
- reg2
- and_inst

次のサンプル・ピン名が使用されます。

- data1|combout
- reg1|regout
- and_inst|combout

次のネット名が使用されます。

- data1~combout
- reg1
- and_inst

次のポート名が使用されます。

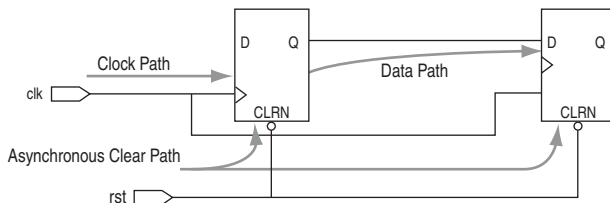
- data1,clk
- data_out

あるレジスタの出力を別のレジスタの入りに接続するなど、パスは2つのデザイン・ノードを接続します。タイミング・パスはタイミング解析において重要な役割を果たします。タイミング・クロージャおよび最適化のために、タイミング・パスのタイプを理解することが重要です。次に示すのは、通常解析されるパスです。ここでは、これらのパスについて説明します。

- エッジ・パス — ポートからピンの間、ピンからピンの間、およびピンからポートの間の接続。
- クロック・パス — デバイス・ポートまたは内部で生成されたクロック・ピンから、レジスタのクロック・ピンまでのエッジ。
- データ・パス — ポートまたはシーケンシャル・エレメントのデータ出力ピンから、ポートまたは別のシーケンシャル・エレメントのデータ入力ピンまでのエッジ。
- 非同期パス — ポートまたはシーケンシャル・エレメントから、シーケンシャル・エレメントの非同期セットまたはクリア・ピンまでのエッジ。

図 7-5 に、これらの通常解析されるパスの種類をいくつか示します。

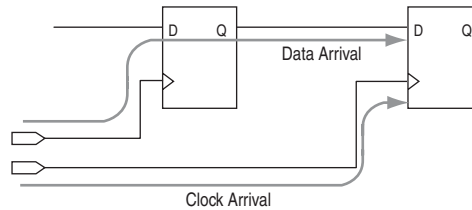
図 7-5. パスの種類



Quartus II TimeQuest タイミング・アナライザは、パスの種類を識別すると、有効なレジスタ間パスのデータ到達時間とクロック到達時間をレポートできます。Quartus II TimeQuest タイミング・アナライザは、クロック・ソースからソース・レジスタのクロック・ピンまでの遅延、ソース・レジスタの clock-to-out (μt_{CO})、およびソース・レジスタの Q ピンからデスティネーション・レジスタ D ピンまでの遅延を加算することにより、データ到達時間を計算します。ここで、 μt_{CO} は FPGA の内部レジスタに固有の clock-to-out です。

Quartus II TimeQuest タイミング・アナライザは、クロック・ソースからデスティネーション・レジスタのクロック・ピンまでの遅延を加算することにより、クロック到着時間を計算します。図 7-6 にデータ到達パスとクロック到達パスを示します。Quartus II TimeQuest タイミング・アナライザは、デスティネーション・レジスタのクロック到達時間とマイクロ・セットアップ時間 (μt_{SU}) を考慮して、データの所要時間を計算します。ここで、 μt_{SU} は FPGA の内部レジスタに固有のセットアップ時間です。

図 7-6. データ到達およびクロック到達

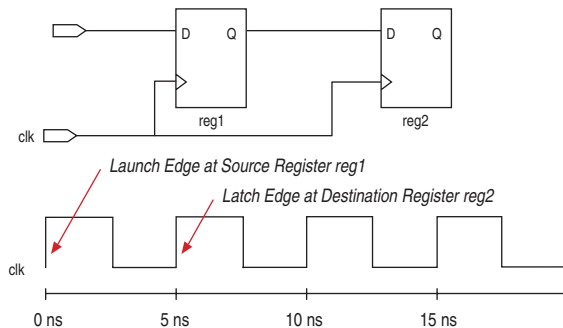


Quartus II TimeQuest タイミング・アナライザは、デザイン内の様々なパスを識別するほか、クロック特性を解析してレジスタ間パスのワースト・ケース値を計算します。デザインのすべてのクロックを制約してから、この解析を実行する必要があります。

ローンチ・エッジは、シーケンシャル・エレメントからデータを送信するアクティブ・クロック・エッジで、データの転送元として動作します。ラッチ・エッジは、シーケンシャル・エレメントのデータ・ポートでデータをキャプチャするアクティブ・クロック・エッジで、データの転送先として動作します。

図 7-7 に、連続するクロック・エッジを使用してデータの転送とキャプチャを行うシングル・サイクル・システム、レジスタ間パス、および対応するローンチ・エッジおよびラッチ・エッジのタイミング図を示します。この例では、ローンチ・エッジは、レジスタ reg1 から 0 ns でデータを送信し、レジスタ reg2 のラッチ・エッジは、5 ns でデータをキャプチャします。

図 7-7. ローンチ・エッジとラッチ・エッジ



Quartus II TimeQuest タイミング・アナライザは、ローンチ・エッジとラッチ・エッジを基準にするクロックのセットアップおよびホールド要件を検証します。

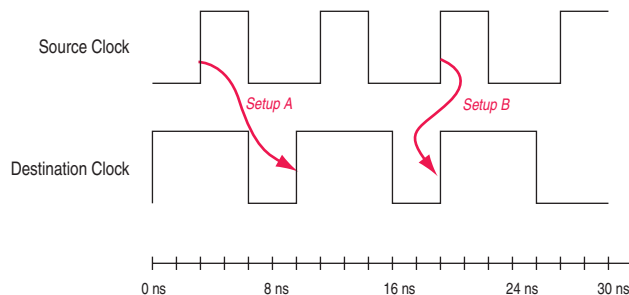
クロック解析

包括的なスタティック・タイミング解析には、レジスタ間パス、I/O パス、および非同期リセット・パスの解析が含まれます。Quartus II TimeQuest タイミング・アナライザは、データ要求時間、データ到達時間、およびクロック到達時間を使用して、回路性能を検証し、発生する可能性があるタイミング違反を検出します。Quartus II TimeQuest タイミング・アナライザは、デザインが正しく機能するために満足する必要があるタイミング関係を決定し、到達時間を所要時間と照合してタイミングを検証します。

クロック・セットアップ・チェック

クロック・セットアップ・チェックを実行するために、Quartus II TimeQuest タイミング・アナライザは、各レジスタ間パスのローンチ・エッジとラッチ・エッジを個々に解析することにより、セットアップ関係を決定します。デスティネーション・レジスタの各ラッチ・エッジについて、Quartus II TimeQuest タイミング・アナライザは、ソース・レジスタの直近のクロック・エッジをローンチ・エッジとして使用します。図 7-8 では、2つのセットアップ関係を定義して、セットアップ A、セットアップ B というラベルを付けています。10 ns におけるラッチ・エッジの場合、ローンチ・エッジとして動作する直近のクロックは、3 ns のクロックで、セットアップ A というラベルが付いています。20 ns におけるラッチ・エッジの場合、ローンチ・エッジとして動作する直近のクロックは、19 ns のクロックで、セットアップ B というラベルが付いています。

図 7-8. セットアップ・チェック



Quartus II TimeQuest タイミング・アナライザは、クロック・セットアップ・チェックの結果をスラック値としてレポートします。スラックとは、タイミング要求を満たすかどうかの境目となる指標です。正のスラックは、タイミング要求を満たすことを示し、負のスラックは、タイミング要求を満たさないことを示します。Quartus II TimeQuest タイミング・アナライザは、**計算式 1** に示す式から、内部レジスタ間パスのクロック・セットアップ・スラックを決定します。

$$(1) \quad \text{Clock Setup Slack} = \text{Data Required Time} - \text{Data Arrival Time}$$

$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay to Source Register} + \mu t_{CO} + \text{Register-to-Register Delay}$$

$$\text{Data Required} = \text{Clock Arrival Time} - \mu t_{SU} - \text{Setup Uncertainty}$$

$$\text{Clock Arrival Time} = \text{Latch Edge} + \text{Clock Network Delay to Destination Register}$$

データ・パスが入力ポートから内部レジスタまでの場合、Quartus II TimeQuest タイミング・アナライザは、**計算式 2** に示す式を使用して、セットアップ・スラック時間を計算します。

$$(2) \quad \text{Clock Setup Slack Time} = \text{Data Required Time} - \text{Data Arrival Time}$$

$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay} + \text{Input Maximum Delay of Pin} + \text{Pin-to-Register Delay}$$

$$\text{Data Required Time} = \text{Latch Edge} + \text{Clock Network Delay to Destination Register} - \mu t_{SU}$$

データ・パスが内部レジスタから出力ポートまでの場合、Quartus II TimeQuest タイミング・アナライザは、**計算式 3** に示す式を使用して、セットアップ・スラック時間を計算します。

$$(3) \quad \text{Clock Setup Slack Time} = \text{Data Required Time} - \text{Data Arrival Time}$$

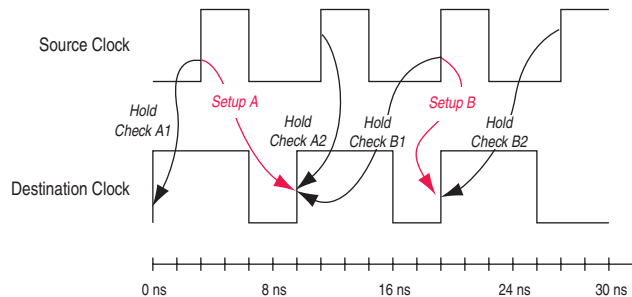
$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay to Source Register} + \mu t_{CO} + \text{Register-to-Pin Delay}$$

$$\text{Data Required Time} = \text{Latch Edge} + \text{Clock Network Delay} - \text{Output Maximum Delay of Pin}$$

クロック・ホールド・チェック

クロック・ホールド・チェックを実行するために、Quartus II TimeQuest タイミング・アナライザは、ソース・レジスタとデスティネーション・レジスタのすべてのペアで存在する、各セットアップ関係に対するホールド関係を決定します。Quartus II TimeQuest タイミング・アナライザは、すべてのセットアップ関係からすべての隣接クロック・エッジをチェックして、ホールド関係を決定します。Quartus II TimeQuest タイミング・アナライザは、各セットアップ関係に対して、2つのホールド・チェックを実行します。最初のホールド・チェックでは、現在のローンチ・エッジによって開始するデータが、前のラッチ・エッジによってキャプチャされないことを判定します。2番目ホールド・チェックでは、次のローンチ・エッジによって開始するデータが、現在のラッチ・エッジによってキャプチャされないことを判定します。図 7-9 に、セットアップ A、セットアップ B のラベルが付いた2つのセットアップ関係を示します。最初のホールド・チェックには、セットアップ A とセットアップ B に対して、それぞれホールド・チェック A1 とホールド・チェック B1 のラベルが付けられています。2番目のホールド・チェックには、セットアップ A とセットアップ B に対して、それぞれホールド・チェック A2 とホールド・チェック B2 のラベルが付けられています。

図 7-9. ホールド・チェック



可能なホールド関係の中から、Quartus II TimeQuest タイミング・アナライザは、最も制限されるホールド関係を選択します。ラッチ・エッジとローンチ・エッジ間の差（すなわち、ラッチ・ローンチ間で、ラッチおよびローンチの絶対値ではない）が最大のホールド関係が選択されます。その理由は、この関係によってレジスタ間パスの許容最小遅延が決定されるためです。図 7-9 の場合、選択されるホールド関係はホールド・チェック A2 です。

Quartus II TimeQuest タイミング・アナライザは、[計算式 4](#) に示すとおり、クロック・ホールド・スラックを算出します。

- (4) Clock Hold Slack = Data Arrival Time – Data Required Time

$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay to Source Register} + \mu t_{C0} + \text{Register-to-Register Delay}$$

$$\text{Data Required Time} = \text{Clock Arrival Time} + \mu t_H + \text{Hold Uncertainty}$$

$$\text{Clock Arrival Time} = \text{Latch Edge} + \text{Clock Network Delay to Destination Register}$$

データ・パスが入力ポートから内部レジスタまでの場合、Quartus II TimeQuest タイミング・アナライザは、[計算式 5](#) に示す式を使用して、ホールド・スラック時間を計算します。

- (5) Clock Setup Slack Time = Data Arrival Time – Data Required Time

$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay} + \text{Input Minimum Delay of Pin} + \text{Pin-to-Register Delay}$$

$$\text{Data Required Time} = \text{Latch Edge} + \text{Clock Network Delay to Destination Register} + \mu t_H$$

データ・パスが内部レジスタから出力ポートまでの場合、Quartus II TimeQuest タイミング・アナライザは、[計算式 6](#) に示す式を使用して、セットアップ・ホールド時間を計算します。

- (6) Clock Setup Slack Time = Data Arrival Time – Data Required Time

$$\text{Data Arrival Time} = \text{Latch Edge} + \text{Clock Network Delay to Source Register} + \mu t_{C0} + \text{Register-to-Pin Delay}$$

$$\text{Data Required Time} = \text{Latch Edge} + \text{Clock Network Delay} - \text{Output Minimum Delay of Pin}$$

リカバリおよびリムーバル

リカバリ時間とは、非同期コントロール信号(例えば、clear や preset)のディアサーションが次のアクティブ・クロック・エッジまで安定していなければならない最小時間です。リカバリ・スラック時間の計算は、クロック・セットアップ・スラック時間の計算に似ていますが、非同期コントロール信号に適用される点が異なります。非同期コントロール信号がラッチされる場合、Quartus II TimeQuest タイミング・アナライザは、[計算式 7](#)に示す式を使用して、リカバリ・スラック時間を計算します。

$$(7) \quad \text{Recovery Slack Time} = \text{Data Required Time} - \text{Data Arrival Time}$$

$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay to Source Register} + \mu t_{CO} + \text{Register-to-Register Delay}$$

$$\text{Data Required Time} = \text{Latch Edge} + \text{Clock Network Delay to Destination Register} - \mu t_{SU}$$

非同期コントロールがラッチされない場合、Quartus II TimeQuest タイミング・アナライザは、[計算式 8](#)に示す式を使用して、リカバリ・スラック時間を計算します。

$$(8) \quad \text{Recovery Slack Time} = \text{Data Required Time} - \text{Data Arrival Time}$$

$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay} + \text{Maximum Input Delay} + \text{Port-to-Register Delay}$$

$$\text{Data Required Time} = \text{Latch Edge} + \text{Clock Network Delay to Destination Register Delay} - \mu t_{SU}$$



非同期リセット信号がポート (デバイス I/O) からの信号である場合、非同期リセット・ポートに Input Maximum Delay アサインメントを行って、Quartus II TimeQuest タイミング・アナライザで、このパスのリカバリ解析を実行できなければなりません。

リムーバル時間とは、非同期コントロール信号のディアサーションがアクティブ・クロック・エッジ後に安定していなければならない最小時間です。Quartus II TimeQuest タイミング・アナライザによるリムーバル時間スラックの計算は、クロック・ホールド・スラック時間の計算に似ていますが、非同期コントロール信号に適用される点が異なります。非同期コントロールがラッチされる場合、Quartus II TimeQuest タイミング・アナライザは、[計算式 9](#)に示す式を使用して、リムーバル・スラック時間を計算します。

$$(9) \quad \text{Removal Slack Time} = \text{Data Arrival Time} - \text{Data Required Time}$$

$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay to Source Register} + \mu t_{CO} \text{ of Source Register} + \text{Register-to-Register Delay}$$

$$\text{Data Required Time} = \text{Latch Edge} + \text{Clock Network Delay to Destination Register} + \mu t_H$$

非同期コントロールがラッチされない場合、Quartus II TimeQuest タイミング・アナライザは、計算式 10 に示す式を使用して、リムーバル・スラック時間を計算します。

$$(10) \quad \text{Removal Slack Time} = \text{Data Arrival Time} - \text{Data Required Time}$$

$$\text{Data Arrival Time} = \text{Launch Edge} + \text{Clock Network Delay} + \text{Input Minimum Delay of Pin} + \text{Minimum Pin-to-Register Delay}$$

$$\text{Data Required Time} = \text{Latch Edge} + \text{Clock Network Delay to Destination Register} + \mu t_H$$



非同期リセット信号がデバイス・ピンからの信号である場合、非同期リセット・ピンに **Input Minimum Delay** 制約を指定して、Quartus II TimeQuest タイミング・アナライザがこのパスでリムーバル解析を実行できるようにする必要があります。

マルチサイクル・パス

マルチサイクル・パスとは、デスティネーション・レジスタでデータをラッチするために 2 つ以上のクロック・サイクルを必要とするデータ・パスのことです。例えば、1 個のレジスタで、2 番目または 3 番目の立ち上がりエッジごとにデータをキャプチャすることが必要な場合があります。図 7-10 に、乗算器の入力レジスタと、デスティネーションが 1 つおきのクロック・エッジでデータをラッチする出力レジスタとの間のマルチサイクル・パスの例を示します。

図 7-10. マルチサイクル・パスの例を示す図

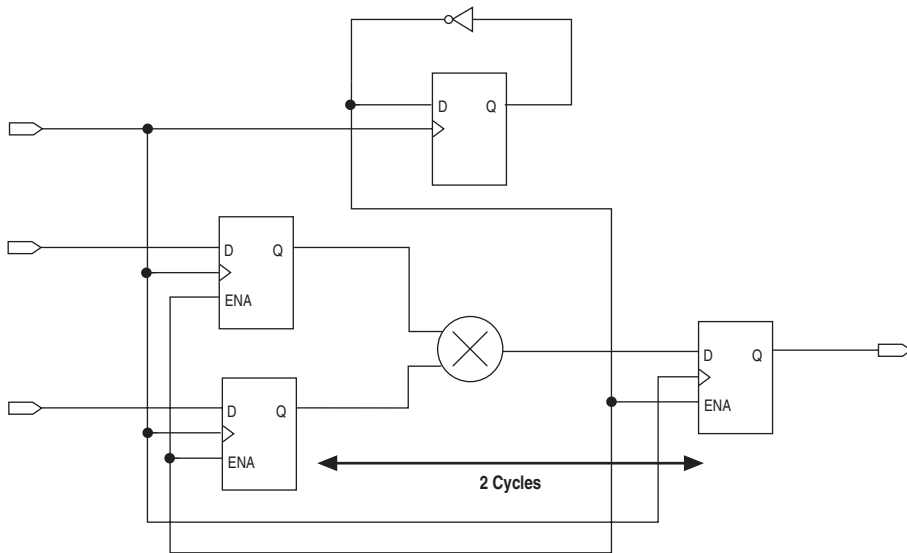


図 7-11 に、レジスタ間パスを示します。ここで、ソース・クロック `src_clk` の周期は 10 ns、デスティネーション・クロック `dst_clk` の周期は 5 ns です。

図 7-11. レジスタ間パス

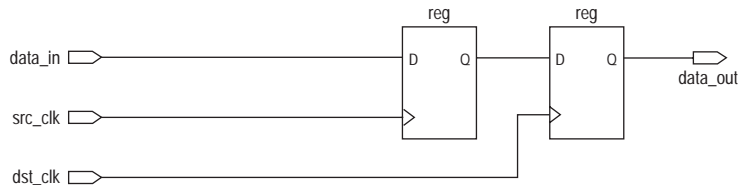
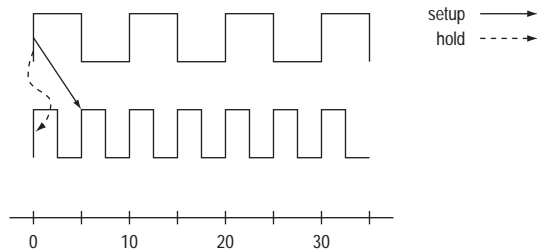


図 7-12 に、ソース・クロックおよびデスティネーション・クロックのタイミング図と、デフォルトのセットアップ関係およびホールド関係を示します。デフォルトのセットアップ関係は 5 ns、デフォルトのホールド関係は 0 ns です。

図 7-12. デフォルトのセットアップおよびホールドのタイミング図



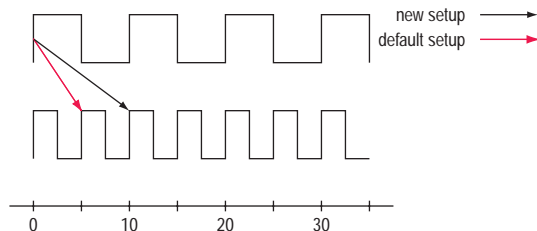
デフォルトのセットアップおよびホールド関係は、システム要件を満たすように `set_multicycle_path` コマンドで修正できます。

表 7-3 に、Quartus II TimeQuest タイミング・アナライザがセットアップ関係またはホールド関係を決定するのに使用する、ローンチ・エッジ時間またはラッチ・エッジ時間を修正するコマンドを示します。

コマンド	修正の説明
<code>set_multicycle_path -setup -end</code>	セットアップ関係のラッチ・エッジ時間
<code>set_multicycle_path -setup -start</code>	セットアップ関係のローンチ・エッジ時間
<code>set_multicycle_path -hold -end</code>	ホールド関係のラッチ・エッジ時間
<code>set_multicycle_path -hold -start</code>	ホールド関係のローンチ・エッジ時間

図 7-13 に、2 マルチサイクル・セットアップを適用した後のタイミング図を示します。このコマンドにより、ラッチ・エッジ時間をデフォルトの 5 ns から 10 ns に移動します。

図 7-13. 修正後のセットアップ図



メタスタビリティ

すべてのレジスタで、入力でデータをキャプチャ可能なセットアップおよびホールド時間要件が定義されています。キャプチャされたデータはレジスタの出力に送られ、入力でキャプチャされた信号に応じて、High または Low 電圧信号が出力されます。ただし、データがセットアップ時間やホールド時間に違反している場合、レジスタの出力は準安定状態になることがあります。この状態では、レジスタの出力は High 状態と Low 状態の間の値で揺らぎます。この値が回路に伝播するとレジスタが不正な値をラッチして、システムに障害が発生する原因となります。

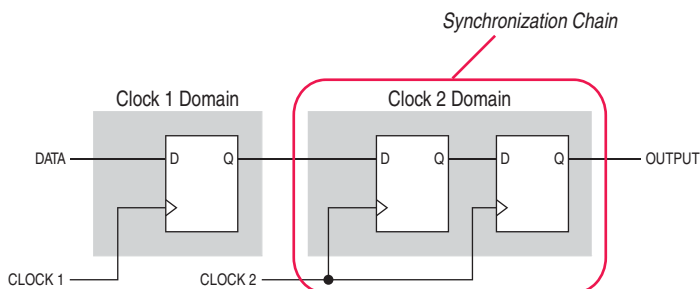
メタスタビリティの問題は一般に、無関係なクロック・ドメインにある 2 組の回路の間でデータ信号が転送されると発生します。メタスタビリティのために発生する障害を少なくするために、回路設計者は一般に、デスティネーション・クロック・ドメインで一連のバック・トゥー・バック・レジスタ（同期レジスタ・チェーン）を使用して、データ信号を新しいクロック・ドメインに再同期します。

同期レジスタ・チェーンは、以下の要件を満たす一連のレジスタとして定義できます。

- レジスタはすべて、同じクロックまたは位相関連クロックによって駆動される。
- 最初のレジスタは、関連性のないクロック・ドメインから（つまり、非同期で）ドライブされ、レジスタのファンアウトは 1 である。

図 7-14 に、同期レジスタ・チェーンのサンプルを示します。

図 7-14. 同期レジスタ・チェーン



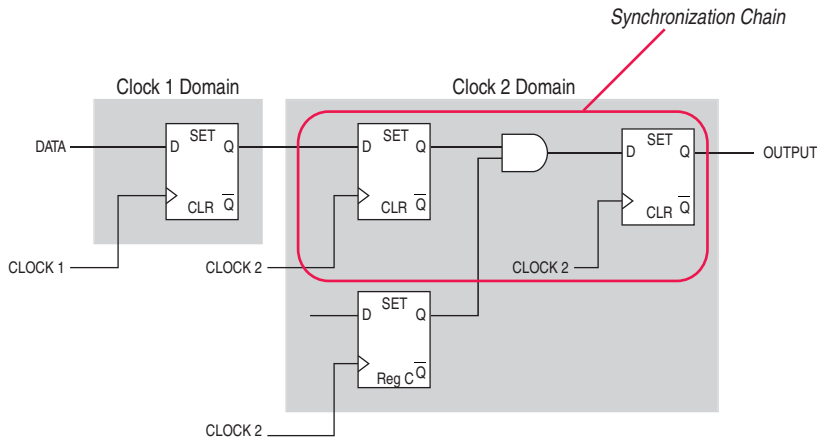
同期レジスタ・チェーンの長さは、上記の要件を満たすこの同期クロック・ドメインのレジスタ数によって定義されます。



同期クロック・ドメイン内の各ネットに有効なセットアップ・スラックがある限り、これらの同期レジスタ間にロジックが存在することができます。

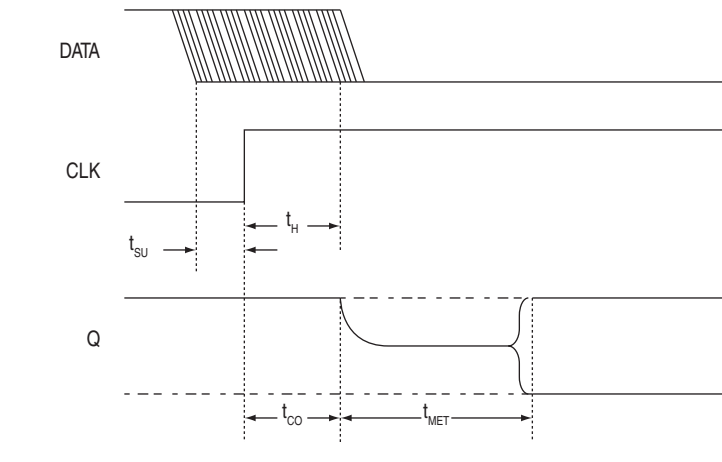
同期レジスタ・チェーンとみなされる例を図 7-15 に示します。

図 7-15. 同期レジスタ・チェーン



メタスタビリティが発生すると、clock-to-output 時間がレジスタの標準 t_{CO} 時間を超えることがあります。出力信号が既知状態に落ち着くのに要する時間のうち、 t_{CO} を超過した時間はセトリング時間 (t_{MET}) と呼ばれます。同期レジスタ・チェーンの t_{MET} は、チェーン内のレジスタのすべての出力スラックを合計した時間です。図 7-16 に、メタステーブル信号の t_{MET} を示します。メタステーブル・イベントの解決に要する時間が、同期チェーンのセトリング時間よりも長い場合、このメタステーブル・イベントが前方に伝播し、デザインに障害が発生すること可能性があります。

図 7-16. メタスタビリティ・タイミング・パラメータ



計算された t_{MET} に基づき、[計算式 11](#) に示す式を使用して、同期チェインの平均故障間隔 (MTBF) を計算できます。

$$(11) \quad MTBF = \frac{e^{(C_2 \times t_{MET})}}{C_1 \times f_{CLOCK} \times f_{DATA}}$$

MTBF は故障が発生してから次に故障が発生するまでの平均時間の概算値です。定数 **C1** および **C2** は、デバイスの製造時に使用されるプロセス技術によって決定され、TimeQuest タイミング・アナライザが適切なデバイスに対して指定します。

TimeQuest タイミング・アナライザは、デザイン内の各同期レジスタ・チェインの MTBF を計算することにより、デザインのメタスタビリティに対する堅牢性を解析します。次に、デザイン全体の MTBF が、デザイン内の同期チェインに基づいて見積もられます。

デザイン内の同期レジスタ・チェインをレポートするほか、Quartus II ソフトウェアは、レジスタの重複やロジック・リタイミングなど、MTBF に悪影響を及ぼす可能性がある最適化からこれらのレジスタを保護します。また、Quartus II ソフトウェアは、デザインの MTBF が低すぎる場合にもそれを最適化することができます。

TimeQuest タイミング・アナライザで、メタスタビリティ解析を有効にして、メタスタビリティをレポートする方法については、[7-70 ページの「report_metastability」](#)を参照してください。



TimeQuest で、メタスタビリティ解析を有効にして、メタスタビリティをレポートする方法については、「Quartus II ハンドブック Volume 2」の「面積およびタイミングの最適化」の章を参照してください。

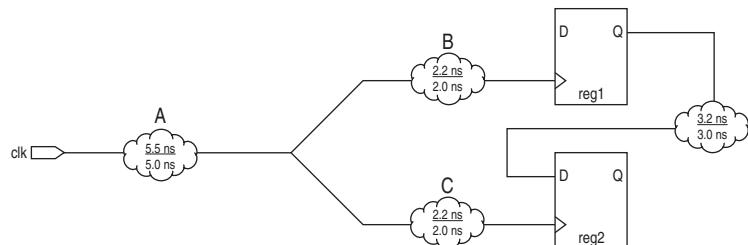
コモン・クロック・パス・ペシミズム

コモン・クロック・パス・ペシミズム (CCPP) リムーバルは、スタティック・タイミング解析で、コモン・クロック・パスに関連する最小遅延と最大遅延の変動を考慮します。CCPP リムーバルは、この変動を考慮するために、共通クロックパスの最大遅延と最小遅延の差を適切なスラック式に追加します。

最小遅延と最大遅延の変動は、同じクロック・パスに異なる 2 つの遅延値を使用すると発生する可能性があります。例えば、簡単なセットアップ解析では、データ到達時間はソース・レジスタへの最大クロック・パス遅延を使用して求められます。また、データ所要時間はデスティネーション・レジスタへの最小クロック・パス遅延を使用して求められます。ただし、ソース・レジスタへのクロック・パスとデスティネーション・レジスタへのクロック・パスで、クロック・パスを共有する場合、解析では最大遅延と最小遅延の両方を使用して、共通クロックパスをモデル化します。2 つの異なる遅延値（最小遅延と最大遅延）を使用して同じクロック・パスをモデル化することはできないため、結果として、過度に悲観的な解析になります。

図 7-17 に、標準的なレジスタ間パスを、最大遅延値と最小遅延値と併せて示します。

図 7-17. 共通クロックパス



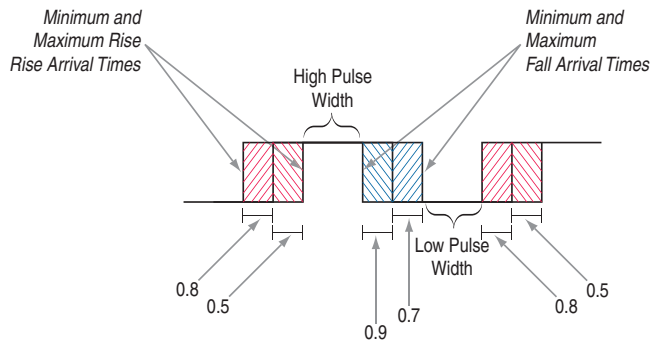
セグメント A は reg1 と reg2 の間の共通クロックパスです。最小遅延は 5.0 ns、最大遅延は 5.5 ns です。最大遅延値と最小遅延値の差は、CCPP リムーバル値と同じです（このケースでは、CCPP は 0.5 ns）。次に、この CCPP リムーバル値は適切なスラック式に追加されます。したがって、

図 7-17 に示すレジスタ間のセットアップ・スラックが CCPP リムーバルなしで 0.7 ns の場合、このスラックは CCPP リムーバル付きで 1.2 ns になります。

CCPP は、レジスタの最小パルス幅を求める場合にも使用されます。クロック信号がレジスタによって認識されるには、レジスタの最小パルス幅要件を満たしている必要があります。最小 High タイムにより、ポジティブ・エッジ・トリガ・レジスタの最小パルス幅が定義されます。最小 Low タイムにより、ネガティブ・エッジ・トリガ・レジスタの最小パルス幅が定義されます。

クロック・パルスがレジスタの最小パルス幅に違反していると、レジスタのデータ・ピンでデータがラッチされません。最小パルス幅のスラックを計算するには、必要な最小パルス幅時間を実際の最小パルス幅時間から減算します。実際の最小パルス幅時間は、レジスタのクロック・ポートに供給されるクロックに指定されるクロック要件によって決まります。必要な最小パルス幅時間は、最大立ち上がり時間、最小立ち上がり時間、最大立ち下がり時間、および最小立ち下がり時間によって決まります。図 7-18 に、High パルスと Low パルスの両方について、必要な最小パルス幅時間の図を示します。

図 7-18. 必要な最小パルス幅



CCPP を使用すると、最大立ち上がり時間から最小立ち上がり時間を減算した時間と、最大立ち下がり時間から最小立ち下がり時間を減算した時間のうち、小さいほうの時間だけ最小パルス幅スラックを増やすことができます。図 7-18 の場合、スラック値は 0.2 ns だけ増やすことができます。この 0.2 ns は、0.3 ns (0.8 ns - 0.5 ns) と 0.2 ns (0.9 ns - 0.7 ns) のうち小さいほうの値です。

TimeQuest タイミング・アナライザで、CCPP をレポートする方法について詳しくは、7-75 ページの「[report_min_pulse_width](#)」を参照してください。

Enable common clock path pessimism removal オプションを使用して、フィッタおよびタイミング解析で CCPP を考慮する必要があります。**Settings** メニューのこのオプションには、**TimeQuest settings** をクリックしてアクセスします。

また、タイミング解析で CCPP を考慮するには、TimeQuest コンソール・ペインで、Tel プロンプトまたはスクリプトに次のように入力します。

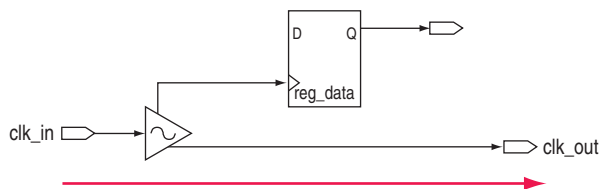
```
enable_ccpp_removal
```

Clock-As-Data

大部分の FPGA デザインには、データ・パスまたはクロック・パスと呼ばれる任意の 2 つのノード間に簡易な接続があります。データ・パスとは、ある同期要素の出力から別の同期要素の入力までの接続です。クロックは同期要素のクロック・ピンへの接続です。ただし、ソース同期インタフェースを使用するなど、FPGA デザインはより複雑になっているため、このような簡単な図では十分に説明できません。

ポート `clk_in` とポート `clk_out` の間の接続は、クロック・パスまたはデータ・パスのいずれとしても扱えます。クロック・パスは、ポート `clk_in` からレジスタ `reg_data` クロック・ピンまでのパスです。データ・パスは、ポート `clk_in` からポート `clk_out` までのパスです。[図 7-19](#) に示すデザインの場合、ポート `clk_in` からポート `clk_out` までのパスは、クロック・パスでありデータ・パスでもあります。

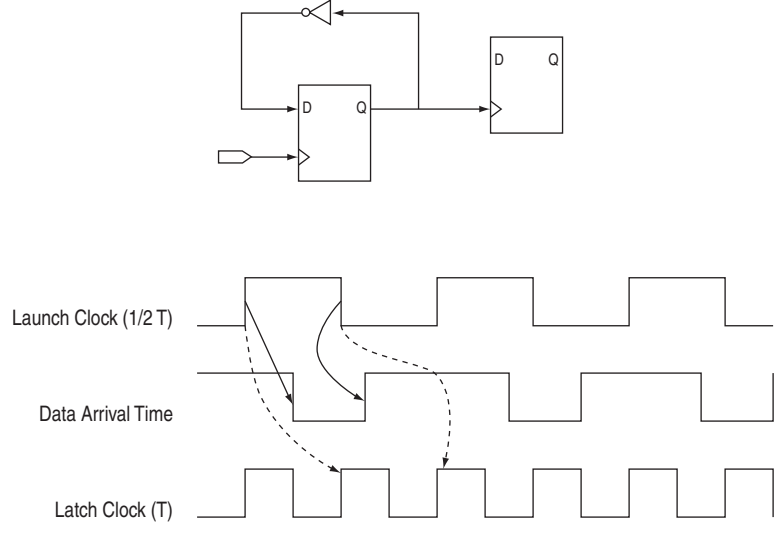
図 7-19. 簡略化されたソース同期出力



Clock-As-Data 解析により、TimeQuest タイミング・アナライザはユーザー制約に基づいて、より精密なパス解析を行うことができます。クロック・パス解析の場合、PLL に関連付けられている位相シフトが考慮されます。データ・パスの場合、PLL に関連付けられている位相シフトが無視されずに考慮されます。

また、Clock-As-Data 解析は、[図 7-20](#) に示すような内部生成されるクロック・ディバイダにも適用されます。

図 7-20. クロック・ディバイダ

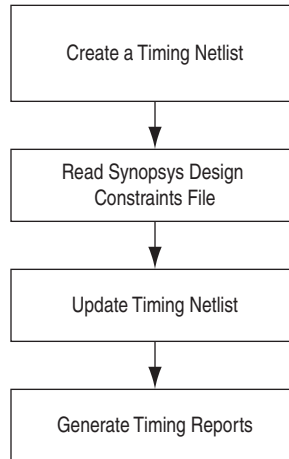


ソース同期インターフェースには、データ信号と並行して伝達されるクロック信号があります。クロックとデータのペアは、同じデバイスで発信されるかまたは終了します。

Quartus II TimeQuest タイミング・ アナライザ のフロー に関する ガイドライン

図 7-21 に示すステップを使用して、TimeQuest タイミング・アナライザでタイミングを検証します。

図 7-21. TimeQuest タイミング・アナライザでのタイミング検証



以下の項では、図 7-21 に示す各ステップについて説明します。

タイミング・ネットリストの作成

フル・コンパイルの実行後、フィッティング後の結果からフル・アノテーションされたデータベースに基づいて、タイミング・ネットリストを作成する必要があります。

タイミング・ネットリストを作成するには、**Tasks** ペインの **Create Timing Netlist** をダブルクリックするか、**Console** ペインで次のコマンドを入力します。

```
create_timing_netlist .J
```

Synopsys Design Constraints ファイルの読み出し

タイミング・ネットリストを作成したら、次は SDC ファイルを読み出す必要があります。このステップでは、SDC ファイルに定義されている制約と例外をすべて読み出します。

SDC ファイルは、**Task** ペインまたは **Console** ペインから読み出すことができます。

Tasks ペインから SDC ファイルを読み出すには、**Read SDC File** コマンドをダブルクリックします。



Read SDC File タスクは、<current revision>.sdc ファイルを読み出します。

Console ペインから SDC ファイルを読み出すには、**Console** で次のコマンドを入力します。

```
read_sdc ↓
```

TimeQuest タイミング・アナライザで SDC ファイルを読み出す方法について詳しくは、7-32 ページの「**Synopsys Design Constraints ファイルの優先順位**」を参照してください。

タイミング・ネットリストの更新

SDC ファイルを読み出したら、タイミング・ネットリストを更新する必要があります。**TimeQuest** タイミング・アナライザは、検証のためにすべての制約をネットリストに適用し、デザイン内の無効なパスやフォルス・パスをすべて検証から除外します。

タイミング・ネットリストを更新するには、**Tasks** ペインの **Update Timing Netlist** をダブルクリックするか、**Console** ペインで次のコマンドを入力します。

```
update_timing_netlist ↓
```

タイミング・レポートの生成

デザイン内のすべてのクリティカル・パスについてタイミング・レポートを生成できます。**Tasks** ペインには、一般的に使用するレポート用コマンドがあります。デザインに関する個別レポートやカスタム・レポートを生成できます。

レポートについて詳しくは、7-66 ページの「**タイミング・レポート**」の項を参照してください。



提供されているレポート API (Application Program Interface) の全リストについては、「**SDC & TimeQuest API Reference Manual**」を参照してください。

タイミングを検証していると、クリティカル・パスに沿ってエラーが発生することがあります。この場合は、既存の制約を改良するか、新しい制約を作成して、既存の制約の効果を変更することができます。制約を修正、削除、または追加した場合は、フル・コンパイルを実行する必要があります。これにより、フィッパは新しい制約に基づいてデザインを再び最適化し、プロセスのコンパイル実行ステップに戻します。この繰り返しプロセスにより、デザインのタイミング違反を解決することができます。



タイミング解析フローを自動化するサンプル Tcl スクリプトについては、「[TimeQuest Quick Start Tutorial](#)」を参照してください。

コレクション

Quartus II TimeQuest タイミング・アナライザは、デザイン内のポート、ピン、セル、またはノードへのアクセスを容易にするコレクション API をサポートしています。コレクション API は、Quartus II TimeQuest タイミング・アナライザで指定される有効な制約や Tcl コマンドとともに使用します。

表 7-4 に、Quartus II TimeQuest タイミング・アナライザでサポートされるコレクション・コマンドを示します。

コマンド	説明
all_clocks	デザイン内のすべてのクロックのコレクションを返します。
all_inputs	デザイン内のすべての入力ポートのコレクションを返します。
all_outputs	デザイン内のすべての出力ポートのコレクションを返します。
all_registers	デザイン内のすべてのレジスタのコレクションを返します。
get_cells	デザイン内のセルのコレクションを返します。コレクション内のすべてのセル名が指定されたパターンに一致します。ワイルドカードを使用して、複数のセルを同時に選択できます。
get_clocks	デザイン内のクロックのコレクションを返します。set_multicycle_path の -from または -to など、別のコマンドへの引数として使用すると、クロックの各ノードは、コレクション内でクロックで駆動されるすべてのノードを表します。デフォルトでは、特定のノード（クロックの場合でも）をコマンドのターゲットとして使用します。
get_nets	デザイン内のネットのコレクションを返します。コレクション内のすべてのネット名が指定されたパターンと一致します。ワイルドカードを使用して、複数のネットを同時に選択できます。
get_pins	デザイン内のピンをコレクションとして返します。コレクションに含まれるピン名はすべて、指定されたパターンと一致するものです。ワイルドカードを使用して、同時に複数のピンを選択できます。
get_ports	デザイン内のポート（デザインの入力と出力）のコレクションを返します。

表 7-5 に、Quartus II TimeQuest タイミング・アナライザでサポートされる SDC 拡張コレクション・コマンドを示します。

コマンド	説明
get_fanouts <filter>	<filter> を先頭にして、ファンアウト・ノードのコレクションを返します。
get_keepers <filter>	デザイン内のキーパー・ノード（非組み合わせノード）のコレクションを返します。
get_nodes <filter>	デザイン内のノードのコレクションを返します。制約または例外を指定するときは、get_nodes コレクションは使用できません。
get_partitions <filter>	<filter> と一致するパーティションのコレクションを返します。
get_registers <filter>	デザイン内のレジスタのコレクションを返します。
get_fanins <filter>	<filter> を先頭にして、ファンイン・ノードのコレクションを返します。
derive_pll_clocks	PLL の出力に生成されたクロックを自動的に作成します。生成されたクロックのプロパティには、MegaWizard® Plug-In Manager で指定された PLL プロパティが反映されます。
get_assignment_groups <filter>	Assignment (Time) Groups オプションを使用して Quartus 設定ファイル (.qsf) に保存されたキーパー、ポート、またはレジスタのコレクションを返します。
remove_clock <clock list>	<clock list> で指定したクロックのリストを削除します。
set_scc_mode <size>	最大 SCC (Strongly Connected Component) ループ・サイズを設定するか、Quartus II TimeQuest タイミング・アナライザで常に SCC での遅延を見積もるよう強制することができます。
set_time_format	時間単位や小数点以下の桁数を含む、時間フォーマットを設定します。



コレクションについては、SDC ファイルおよび「[SDC and TimeQuest API Reference Manual](#)」を参照してください。

アプリケーション例

例 7-1 に、create_clock および create_generated_clock コマンドのさまざまな使用方法と特定のデザイン構造を示します。

例 7-1. create_clock および set_multicycle_path コマンドと特定のデザイン構造

```
# デューティ・サイクル 60% のシンプルな 10 ns を作成
create_clock -period 10 -waveform {0 6} -name clk [get_ports clk]
# 以下のマルチサイクルは、clk によってクロックされるレジスタで終了する
# すべてのパスに適用
set_multicycle_path -to [get_clocks clk] 2
```

SDC 制約 ファイル

Quartus II TimeQuest タイミング・アナライザは、すべてのタイミング制約を SDC ファイルに格納します。配置配線とタイミング解析について、異なる制約を持つ SDC ファイルを作成できます。



SDC ファイルには、SDC および Tcl コマンドのみ含めます。タイミング・ネットリストを操作するコマンドや、コンパイル・フローを制御するコマンドを SDC ファイルに含めることはできません。

Quartus II ソフトウェアは SDC ファイルを自動的に更新しません。新しい制約や更新された制約は、TimeQuest タイミング・アナライザ GUI で明示的に記述する必要があります。write_sdc コマンドを使用するか、Quartus II TimeQuest タイミング・アナライザで、Constraints メニューの **Write SDC File** をクリックして、SDC ファイルに制約を書き込みます。



SDC ファイルの制約は順序に関係があります。制約は必ず宣言した後で参照しなければなりません。例えば、生成されたクロックが、clk という名前のベース・クロックを参照する場合、ベース・クロックの制約を生成されたクロックの制約より前に宣言する必要があります。

SDC ファイルによるフィッタおよびタイミング解析

Quartus II フィッタによる配置配線、および Quartus II TimeQuest タイミング・アナライザによるスタティック・タイミング解析には、同じ SDC ファイルまたは別の SDC ファイルを指定できます。別の SDC ファイルを使用すると、Quartus II TimeQuest タイミング・アナライザで、配置配線に対する制約セットと、最終的なタイミング・サイン・オフに対する制約セットを持つことができます。

配置配線用 SDC ファイルの指定

フィッタに SDC ファイルを指定するには、Quartus II プロジェクトに SDC ファイルを追加する必要があります。このファイルをプロジェクトに追加するには、Tcl コンソールで次のコマンドを使用します。

```
set_global_assignment -name SDC_FILE <SDC file name>
```

あるいは、Quartus II ソフトウェア GUI で、Project メニューの **Add/Remove Files in Project** をクリックします。

フィッタにより、プロジェクトの SDC ファイルの要件に基づいてデザインが最適化されます。

Compilation Report のタイミング解析レポートに示す結果は、プロジェクトに追加された SDC ファイルに基づきます。



Quartus II TimeQuest タイミング・アナライザをデフォルトのタイミング・アナライザとして指定し、フィッタで SDC ファイルを読み出すようにする必要があります。

スタティック・タイミング解析用 SDC ファイルの指定

Quartus II TimeQuest タイミング・アナライザでタイミング・ネットリストを作成した後、タイミング制約および例外を指定して、タイミング解析を実行できるようにする必要があります。タイミング要求は、フィッタに提供されるものと同じである必要はありません。タイミング要求は手動で指定できます。あるいは、以前に作成された SDC ファイルを読み出すこともできます。

タイミング要求は手動で入力するには、制約入力ダイアログ・ボックスまたは SDC コマンドを使用できます。タイミング要求を含む SDC ファイルがある場合は、そのファイルを使用してタイミング要求を適用します。Quartus II TimeQuest タイミング・アナライザで、タイミング解析用の SDC ファイルを指定するには、次のコマンドを使用します。

```
read_sdc [<SDC file name>]
```

TimeQuest GUI を使用して、タイミング解析に SDC ファイルを適用する場合は、Quartus II TimeQuest タイミング・アナライザで、Constraints メニューの **Read SDC File** をクリックします。



デフォルトでは、Tasks ペインの **Read SDC File** コマンドは、Quartus II 設定ファイル (.qsf) に指定される SDC ファイル (フィッタで使用する SDC ファイルと同じ) を読み出します。

Synopsys Design Constraints ファイルの優先順位

Quartus II フィッタと Quartus II TimeQuest タイミング・アナライザは、SDC ファイルを QSF ファイルのファイル・リストの上から下に順番に読み出します。

Quartus II ソフトウェアは、[図 7-22](#) に示す方法で、SDC ファイルを検索します。

図 7-22. Synopsys Design Constraints ファイルの優先順位

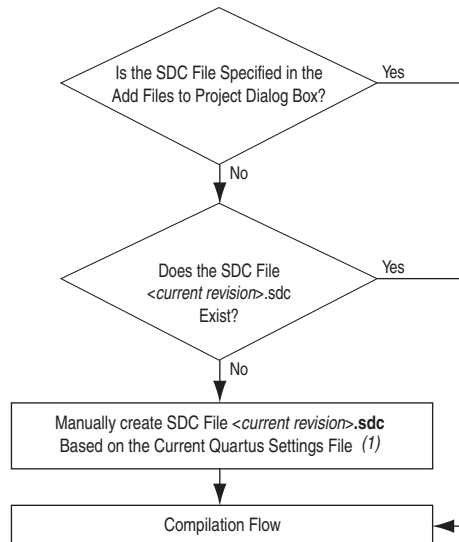



図 7-22 の注:

- (1) これは Quartus II TimeQuest タイミング・アナライザでのみ発生し、Quartus II ソフトウェアでのコンパイル中には発生しません。Quartus II TimeQuest タイミング・アナライザは、起動時に SDC ファイルが存在しない場合、SDC への QSF タイミング・アサインメントの変換を自動化する機能を備えています。

 read_sdc コマンドを引数なしでコマンドラインに入力すると、図 7-22 に示す優先順位に従って処理されます。

クロックの仕様

正確なスタティック・タイミング解析結果を得るには、デザイン内のすべてのクロックおよび関連クロック特性の仕様が不可欠です。Quartus II TimeQuest タイミング・アナライザは、各種クロック方式と任意のクロック特性に対応する多数の SDC コマンドをサポートしています。

この項では、クロック特性を作成および指定するのに使用可能な SDC ファイル API について説明します。

クロック

create_clock コマンドを使用して、任意のレジスタ、ポート、またはピンでクロックを作成します。固有の特性を持つ各クロックを作成できます。例 7-2 に、create_clock コマンドとオプションを示します。

例 7-2. create_clock コマンド

```
create_clock
-period <period value>
[-name <clock name>]
[-waveform <edge list>]
[-add]
<targets>
```

表 7-6 に、create_clock コマンドのオプションを示します。

オプション	説明
-period <period value>	クロック周期を指定します。また、クロック周期は、-period <num>MHz などの周波数単位でも指定できます。(1)
-name <clock name>	特定のクロックの名前 (sysclock など)。クロック名を指定しない場合、クロック名はクロックが割り当てられるノードと同じになります。
-waveform <edge list>	クロックの立ち上がりエッジと立ち下がりエッジを指定します。edge list には、立ち上がりエッジと立ち下がりエッジを交互に指定します。例えば、最初の立ち上がりエッジが 0 ns で発生し最初の立ち下がりエッジが 5 ns で発生する 10 ns 周期の場合、-waveform { 0 5 } と記述します。差は 1 周期単位内であればならず、立ち上がりエッジは立ち下がりエッジの前でなければなりません。デフォルトの edge list は { 0 <period>/2 }、つまりデューティ・サイクル 50% です。
-add	同じポートまたはピンに複数のクロックを指定できます。
<targets>	アサインメントを適用するポートまたはピンを指定します。ソース・オブジェクトが指定されていない場合、クロックが仮想クロックになります。詳細は、7-39 ページの「仮想クロック」を参照してください。

表 7-6 の注：

(1) Quartus II TimeQuest タイミング・アナライザのデフォルト時間単位はナノ秒 (ns) です。

例 7-3 に、デューティ・サイクル 50% の 10 ns のクロックを作成する方法を示します。ここで、最初の立ち上がりエッジはポート clk に印加される 0 ns で発生します。

例 7-3. 100 MHz クロックの作成

```
create_clock -period 10 -waveform { 0 5 } clk
```

例 7-4 に、ポート `clk_sys` に印加される 90 度位相シフトされデューティ・サイクル 50% の 10 ns クロックを作成する方法を示します。

例 7-4. 90 度位相シフトした 100 MHz クロックの作成

```
create_clock -period 10 -waveform { 2.5 7.5 } clk_sys
```

`create_clock` コマンドで定義されるクロックのデフォルト・ソース・レイテンシ値はゼロです。Quartus II TimeQuest タイミング・アナライザは、非仮想クロックに対するクロックのネットワーク・レイテンシを自動的に計算します。

生成クロック

Quartus II TimeQuest タイミング・アナライザは、クロック・ディバイダ、リップル・クロック、または入力されるクロックまたはマスタ・クロックの特性を修正または変更する回路を生成クロックとみなします。これらの回路の出力は、生成クロックとして定義する必要があります。この定義により、Quartus II TimeQuest タイミング・アナライザは、これらのクロックを解析し、関連するネットワーク・レイテンシを考慮できるようにします。

`create_generated_clock` コマンドを使用して、生成クロックを作成します。例 7-5 に、`create_generated_clock` コマンドと使用可能なオプションを示します。

例 7-5. `create_generated_clock` コマンド

```
create_generated_clock
[-name <clock name>]
-source <master pin>
[-edges <edge list>]
[-edge_shift <shift list>]
[-divide_by <factor>]
[-multiply_by <factor>]
[-duty_cycle <percent>]
[-add]
[-invert]
[-master_clock <clock>]
[-phase <phase>]
[-offset <offset>]
<targets>
```

表 7-7 に、create_generated_clock コマンドのオプションを示します。

表 7-7. create_generated_clock コマンドのオプション	
オプション	説明
-name <clock name>	生成クロックの名前。clk_x2 など。クロック名を指定しない場合、クロック名はクロックが割り当てられる最初のノードと同じになります。
-source <master pin>	<master pin> は、クロック設定が派生するデザイン内のノードを指定します。
-edges <edge list> -edge_shift <shift list>	-edges オプションは、マスタ・クロックの立ち上がりおよび立ち下がりエッジに対して新しい立ち上がりおよび立ち下がりエッジを指定します。マスタ・クロックの立ち上がりおよび立ち下がりエッジには、最初の立ち上がりエッジから順に 1.. <i>n</i> の番号が付けられます。例えば、最初の立ち上がりエッジはエッジ 1 です。エッジ 1 の後の最初の立ち下がりエッジは、エッジ番号 2、次の立ち上がりエッジのエッジ番号は 3 と続きます。<edge list> は昇順で指定する必要があります。同じエッジを 2 つのエントリに使用して、元の波形のデューティ・サイクルとは関係なくクロック・パルスを示すことができます。 -edge_shift は、<edge list> の各エッジのシフト量を指定します。-invert オプションを使用すると、-edges と -edge_shifts の印加後にクロックを反転できます。(1)
-divide_by <factor> -multiply_by <factor>	-divide_by および -multiply_by 係数は、クロックの最初の立ち上がりエッジに基づき、指定された係数に従って波形を拡張または縮小します。例えば、-divide_by 2 は -edges {1 3 5} と同じです。通倍クロックの場合は、デューティ・サイクルも指定できます。Quartus II TimeQuest タイミング・アナライザでは、通倍係数と分周係数を同時に指定できます。
-duty_cycle <percent>	生成クロックのデューティ・サイクルを指定します。デューティ・サイクルは最後に適用されます。
-add	同じピンに複数のクロックを指定できます。
-invert	反転はデューティ・サイクルを除くすべての修正が適用された後、クロックの出力で適用されます。
-master_clock <clock>	マスタ・ピンに複数のクロックが存在する場合は、-master_clock を使用してクロックを指定します。
-phase <phase>	生成クロックの位相を指定します。
-offset <offset>	生成クロックのオフセットを指定します。
<targets>	A サインメントを適用するポートまたはピンを指定します。

表 7-7 の注：

(1) Quartus II TimeQuest タイミング・アナライザでは、edge list に最大 3 つのエッジを指定できます。

ソース・レイテンシは、マスタ・クロック（必ずしもマスタ・ピンではない）からのクロック・ネットワーク遅延に基づきます。set_clock_latency -source コマンドを使用して、ソース・レイテンシを無視できます。

図 7-23 に、10 ns クロックに基づいて生成クロックを反転する方法を示します。

図 7-23. 反転クロックの生成

```
create_clock -period 10 [get_ports clk]
create_generated_clock -divide_by 1 -invert -source [get_registers clk] \
  [get_registers gen|clkreg]
```

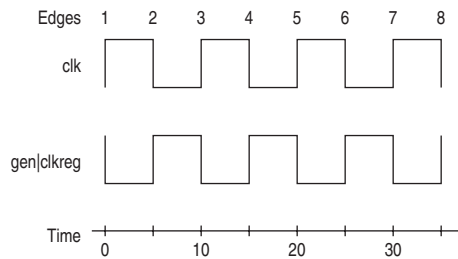


図 7-24 に、`-edges` および `-edge_shift` オプションを使用して、生成クロックを修正する方法を示します。

図 7-24. エッジおよび生成クロックをシフトするエッジ

```
create_clock -period 10 -waveform { 0 5 } [get_ports clk]
# divide-by-2 クロックを作成
create_generated_clock -source [get_ports clk] -edges {1 3 5 } [get_registers \
clkdivA|clkreg]
# マスタ・クロックのデューティ・サイクル（現在 50%）に関係なく、divide-by-2 クロックを作成
create_generated_clock -source [get_ports clk] -edges {1 1 5} -edge_shift { 0 2.5 0 } \
[get_registers clkdivB|clkreg]
```

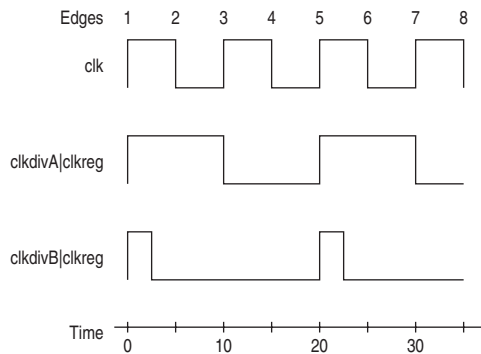
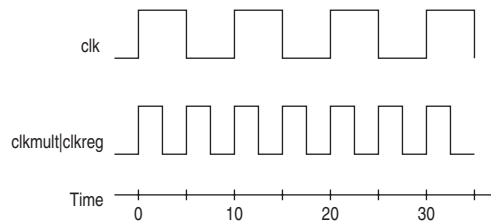


図 7-25 に、`-multiply_by` オプションが生成クロックに与える影響を示します。

図 7-25. 生成クロックの通倍

```
create_clock -period 10 -waveform { 0 5 } [get_ports clk]
# multiply-by-2 クロックを作成
create_generated_clock -source [get_ports clk] -multiply_by 2 [get_registers \
clkmult|clkreg]
```



仮想クロック

仮想クロックとは、デザイン内に実際のソースを持たないクロック、またはデザインと直接作用しないクロックです。例えば、外部デバイスのクロック・ポートには供給されるが、デザインのクロック・ポートには供給されないクロックがあり、デザインのポートに外部デバイスが供給する（または、デザインのポートから外部デバイスに供給される）場合、このようなクロックは仮想クロックとみなされます。

`create_clock` コマンドを使用し、ターゲットを指定しないで仮想クロックを作成します。


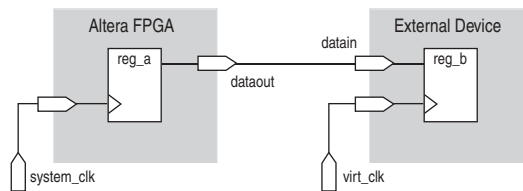
 仮想クロックは、`set_input_delay` および `set_output_delay` 制約に使用できます。

図 7-26 に、Quartus II TimeQuest タイミング・アナライザで、外部レジスタとデザインのレジスタとの関係を適切に解析するために、仮想クロックが必要となる例を示します。オシレータ `virt_clk` は、アルテラ・デバイスとは交信しませんが、外部レジスタのクロック・ソースとして機能するため、クロック `virt_clk` を宣言する必要があります。例 7-6 に、10 ns の仮想クロックを `virt_clk` という名前で作成するコマンドを示します。デューティ・サイクルは 50% で、最初の立ち上がりエッジは 0 ns で発生します。仮想クロックは、出力遅延制約のクロック・ソースとして使用されます。

図 7-26. 仮想クロック・ボード・トポロジー



仮想クロックを作成した後は、アルテラ・デバイスのレジスタと外部デバイスのレジスタとのレジスタ間解析を実行できます。

例 7-6. 仮想クロック例 1

```
# デザインのベース・クロックを作成
create_clock -period 5 [get_ports system_clk]
# 外部レジスタの仮想クロックを作成
create_clock -period 10 -name virt_clk -waveform { 0 5 }
# 仮想クロックを参照する出力遅延を設定
set_output_delay -clock virt_clk -max 1.5 [get_ports dataout]
```

例 7-7 に、デューティ・サイクル 50% で、90° 位相シフトした 10 ns の仮想クロックを作成するコマンドを示します。

例 7-7. 仮想クロックの例 2

```
create_clock -name virt_clk -period 10 -waveform { 2.5 7.5 }
```

マルチ周波数クロック

デザインによっては、1 つのクロック・ポートに複数のクロック・ソースが供給される場合もあります。追加クロックの例として、低消費電力クロックとして動作する、プライマリ・クロックよりも低周波数のクロックがあります。このようなデザインを解析するために、`create_clock` コマンドは `-add` オプションをサポートしており、クロック・ノードに複数のクロックを追加できます。

例 7-8 に、クロック・ポート `clk` に適用する 10 ns のクロックを作成し、さらに同じクロック・ポートに 15 ns のクロックを追加するコマンドを示します。Quartus II TimeQuest タイミング・アナライザは、両方のクロックを使用してタイミング解析を実行します。

例 7-8. マルチ周波数の例

```
create_clock -period 10 -name clock_primary -waveform { 0 5 } [get_ports clk]
create_clock -period 15 -name clock_secondary -waveform { 0 7.5 } [get_ports clk] -add
```

自動クロック検出

デザイン内のすべてのクロック・ノードに対するクロックを自動的に作成するには、`derive_clocks` コマンドを使用します。このコマンドは、ポートまたはレジスタのクロックを作成して、デザイン内のどのレジスタもクロックを持つようにします。

例 7-9 に、`derive_clocks` コマンドとオプションを示します。

例 7-9. `derive_clocks` コマンド

```
derive_clocks
[-period <period value>]
[-waveform <edge list>]
```

表 7-8 に、`derive_clocks` コマンドのオプションを示します。

オプション	説明
<code>-period <period value></code>	クロック周期を作成します。周波数は、 <code>-period <num>MHz</code> として指定することもできます。(1)
<code>-waveform <edge list></code>	クロックの立ち上がりエッジと立ち下がりエッジを作成します。edge list には、立ち上がりエッジと立ち下がりエッジを交互に指定します。例えば、最初の立ち上がりエッジが 0 ns で発生し、最初の立ち下がりエッジが 5 ns で発生する 10 ns 周期の場合、edge list は waveform {0 5} となります。差は 1 周期単位内でなければならない、立ち上がりエッジは立ち下がりエッジの前でなければなりません。デフォルトの edge list は {0 period/2}、つまりデュティ・サイクル 50% です。

表 7-8 の注：

(1) このオプションは、デフォルトの時間単位であるナノ秒 (ns) を使用します。



`derive_clocks` コマンドは PLL の出力にクロックを作成しません。

`derive_clocks` コマンドは、レジスタのクロック・ピンに供給する各レジスタまたはポートに `create_clock` を使用するのと同じです。



最終的なタイミング・サイン・オフに `derive_clocks` コマンドを使用することはお勧めできません。`create_clock` および `create_generated_clock` コマンドを使用して、すべてのクロック・ソースのクロックを作成する必要があります。

派生 PLL クロック

PLL は、アルテラ・デバイスでのクロック管理および合成に使用されます。デザイン要件に基づいて、PLL の出力から生成されたクロックをカスタマイズできます。すべてのクロック・ノードに対してクロックを作成する必要があるため、PLL のすべての出力に関連クロックが必要です。

create_generated_clock コマンドを使用して、PLL の各出力に手動でクロックを作成できます。あるいは、derive_pll_clocks コマンドを使用することもできます。このコマンドは、タイミング・ネットリストを自動的に検索し、各 PLL 出力に指定された設定に応じて、すべての PLL 出力に生成クロックを作成します。

derive_pll_clocks コマンドを使用して、PLL の各出力に自動的にクロックを作成します。例 7-10 に、derive_pll_clocks コマンドとオプションを示します。

例 7-10. derive_pll_clocks コマンド

```
derive_pll_clocks
[-create_base_clocks]
[-use_tan_name]
```

表 7-9 に、derive_pll_clocks コマンドのオプションを示します。

表 7-9. derive_pll_clocks コマンドのオプション	
オプション	説明
-use_tan_name	デフォルトでは、クロック名は出力クロック名です。このオプションは、Quartus II クラシック・タイミング・アナライザで使用する名前に類似したネット名を使用します。
-create_base_clocks	デザインの入力クロック・ポートに、PLL に供給するベース・クロックを作成します。

derive_pll_clocks コマンドは、create_generated_clock コマンドを呼び出して、PLL の出力に生成クロックを作成します。create_generated_clock コマンドのソースは PLL の入力クロック・ピンです。derive_pll_clocks コマンドの発行前または発行後は、PLL の入力クロック・ポートのためのベース・クロックを手動で作成する必要があります。PLL の入力クロック・ノードにクロックが定義されていない場合、PLL 出力についてクロックはレポートされません。代わりに、Quartus II TimeQuest タイミング・アナライザは、タイミング・ネットリストが更新されると、図 7-11 のような警告メッセージを発行します。

例 7-11. 警告メッセージ

```
Warning:The master clock for this clock assignment could not be derived.
Clock:<name of PLL output clock pin name> was not created.
```

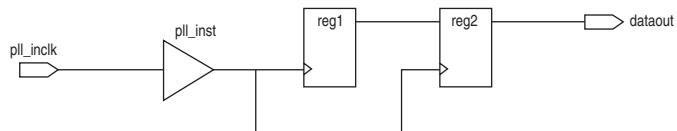
SDCファイルに`derive_pll_clocks`コマンドを含めることができます。これにより、PLL に対するすべての変更を自動的に `derive_pll_clocks` コマンドで検出できます。`derive_pll_clocks` コマンドを SDC ファイルで使用すると、ファイルが読み出されるたびに、PLL 出力クロック・ピンに対する適切な `create_generated_clocks` コマンドが生成されます。`write_sdc -expand` コマンドを `derive_pll_clocks` コマンドの後で使用すると、新しい SDC ファイルには PLL 出力クロック・ピンのために、`derive_pll_clocks` コマンドではなく、個々の `create_generated_clock` コマンドが取り込まれます。PLL のプロパティに対する変更は、新しい SDC ファイルに自動的に反映されません。`derive_pll_clocks` コマンドによって記述される新しい SDC ファイル内の `create_generated_clock` コマンドを手動で更新して、PLL に対する変更を反映させる必要があります。



また、`derive_pll_clocks` 制約は、適切なマルチサイクル制約を追加してパラレル変換係数を考慮することにより、デザイン内のLVDSトランスミッタまたはLVDSレシーバも制約します。

例えば、[図 7-27](#) に、レジスタ間パスを持つ簡単な PLL デザインを示します。

図 7-27. 簡単な PLL デザイン




`derive_pll_clocks` コマンドを使用して、PLL を自動的に制約します。このコマンドが、[図 7-27](#) に示すデザインに対して発行されると、[例 7-12](#) に示すメッセージが生成されます。

例 7-12. `derive_pll_clocks` によって生成されるメッセージ

```


Info:
Info:Deriving PLL Clocks:
Info:create_generated_clock -source pll_inst|altpll_component|pll|inclk[0] -divide_by 2 -name
pll_inst|altpll_component|pll|CLK[0] pll_inst|altpll_component|pll|clk[0]
Info:
  
```

ソース・オプション用のノード名 `pll_inst|altpll_component|pll|inclk[0]` は、PLL の入力クロック・ピンを参照します。また、PLL の出力クロックの名前は、PLL 出力クロック・ノードの名前 `pll_inst|altpll_component|pll|clk[0]` です。

 PLL がクロック・スイッチオーバー・モードの場合、PLL の出力クロックに複数のクロックが作成されます。1 つはプライマリ入力クロック用のクロック（例えば、`inclk[0]`）、もう 1 つはセカンダリ入力クロック用のクロック（例えば、`inclk[1]`）です。この場合、`set_clock_groups` コマンドを `-exclusive` オプションを指定して使用することで、プライマリおよびセカンダリ出力クロックをカットする必要があります。

このデザインについてのレポートを生成するには、まず PLL の入力クロック・ポートにベース・クロックを作成する必要があります。次のようなコマンドを使用します。


```
create_clock -period 5 [get_ports pll_inclk]
```

 `pll_inst|altpll_component|pll|inclk[0]` を使用して、PLL の入力クロック・ピンにベース・クロックを生成する必要はありません。PLL の入力クロック・ポートに作成されるクロックは、PLL の入力クロック・ピンを含め、クロック・ポートのすべてのファンアウトに伝播します。

デフォルトのクロック制約

Quartus II TimeQuest タイミング・アナライザは、完全なクロック解析を提供するために、デザイン内にベース・クロック制約がない場合は、デフォルトによりデザイン内で検出される制約されていないすべてのクロック・ノードに対して自動的にクロックを作成します。Quartus II TimeQuest タイミング・アナライザは、次のコマンドを使用して、制約されていないクロック・ノードに 1 GHz の要件を持つベース・クロックを作成します。

```
derive_clocks -period 1
```

 デザイン内のすべてのクロックに対して、個別クロック制約（例えば、`create_clock` と `create_generated_clock`）を作成する必要があります。これにより、デザインのタイミング要求の完璧で実用的な解析を行うことができます。最終的なタイミング・サイン・オフには、`derive_clocks` を使用しないでください。

デフォルトのクロック制約が適用されるのは、Quartus II TimeQuest タイミング・アナライザがどの同期エレメントにもクロックが関連付けられていないことを検出した場合のみです。例えば、デザインに2つのクロックが含まれており、その一方のクロックにのみ制約がある場合、デフォルトのクロック制約は適用されません。ただし、どちらのクロックも制約されていない場合、デフォルトのクロック制約が適用されます。

クロック・グループ

デザインには多くのクロックが存在できますが、すべてのクロックが互いに交信するわけではなく、交信できないクロックもあります。

set_clock_groups コマンドを使用して、排他的または非同期的なクロックを指定します。例 7-13 に、set_clock_groups コマンドとオプションを示します。

例 7-13. set_clock_groups コマンド

```
set_clock_groups
[-asynchronous | -exclusive]
-group <clock name>
[-group <clock name>]
[-group <clock name>] ...
```

表 7-10 に、set_clock_groups コマンドのオプションを示します。

オプション	説明
-asynchronous	非同期クロック — 2 つのクロック間に位相関係はなく、同時にアクティブになる場合。
-exclusive	排他的クロック — 2 つのクロックのいずれか一方が所定のタイミングでアクティブになる場合。排他的クロック・グループの例として、2 つのクロックが 2-to-1 MUX に供給される場合があります。
-group <clock name>	相互排他的で有効なデスティネーション・クロック名を指定します。クロック名は、<clock name> を使用して指定します。

exclusive オプションは、2 つのクロックが相互排他的で、デザインに同時に存在できないことを宣言する場合に使用します。同じノードに複数のクロックが作成される場合や、マルチプレクス化されたクロックに対して複数のクロックが作成されるときに、このような状況が発生することがあります。例えば、ポート A は 25 MHz または 50 MHz クロックでも供給可能で、このポートに 2 つのクロックを作成できる場合は、set_clock_groups -exclusive を使用して、これらのクロックがデザイ

ンに同時に存在できないことを宣言する必要があります。これにより、25 MHz クロックと 50 MHz クロックの間で生成される可能性があるクロック転送がなくなります。例 7-14 に、この場合の制約を示します。

例 7-14. exclusive オプション

```
create_clock -period 40 -name clk_A [get_ports {port_A}]
create_clock -add -period 20 -name clk_B [get_ports {port_A}]
set_clock_groups -exclusive -group {clk_A} -group {clk_B}
```

グループは、-group オプションで定義されます。TimeQuest タイミング・アナライザは、独立した-group グループのそれぞれのクロック間のタイミング・パスをカットします。

asynchronous オプションは、関連および非関連クロックをグループ化するのに使用します。asynchronous オプションを使用すると、グループに含まれるクロックは互いに非同期とみなされます。各グループ内のクロックは、互いに同期的とみなされます。

例えば、clk_A、clk_B、および clk_C という 3 つのクロックがあるとします。クロック clk_A および clk_B は互いに関連しますが、クロック clk_C は clk_A または clk_B と完全に非同期的に動作します。例 7-15 では、clk_A と clk_B を同じグループ内で関連付け、clk_C を含む 2 番目のグループとは非関連とします。

例 7-15. asynchronous オプションの例 1

```
set_clock_groups -asynchronous -group {clk_A clk_B} -group {clk_C}
```

例 7-16 に、例 7-15 と同じ制約を指定する別の方法を示します。

例 7-16. asynchronous オプションの例 2

```
set_clock_groups -asynchronous -group {clk_C}
```

この場合、clk_C は制約の唯一のグループなので、デザインのクロックと 1 つおきに非関連になります。



TimeQuest タイミング・アナライザは特に制約されていない限り、デフォルトではすべてのクロックが関連付けられていると仮定します。

例 7-17 に、`set_clock_groups` コマンドと、同等の `set_false_path` コマンドを示します。

例 7-17. `set_clock_groups`

クロック A および C は、クロック B および D がアクティブになっているときはアクティブにならない

```
set_clock_groups -exclusive -group {A C} -group {B D}
```

フォルス・パスを使用して同じ内容を指定

```
set_false_path -from [get_clocks A] -to [get_clocks B]
set_false_path -from [get_clocks A] -to [get_clocks D]
set_false_path -from [get_clocks C] -to [get_clocks B]
set_false_path -from [get_clocks C] -to [get_clocks D]
set_false_path -from [get_clocks B] -to [get_clocks A]
set_false_path -from [get_clocks B] -to [get_clocks C]
set_false_path -from [get_clocks D] -to [get_clocks A]
set_false_path -from [get_clocks D] -to [get_clocks C]
```

クロック作用特性

`create_clock` および `create_generated_clock` コマンドは、ボードの影響を考慮しない理想的なクロックを生成します。この項では、クロック・レイテンシとクロック不確実性に関して、クロック作用特性を考慮する方法について説明します。

クロック・レイテンシ

クロック・レイテンシには、ソースとネットワークの 2 種類があります。ソース・レイテンシは、クロックの起点からクロック・デスティネーション・ポイント（例えば、クロック・ポート）までの伝播遅延です。ネットワーク・レイテンシは、クロック定義ポイントからレジスタのクロック・ピンまでの伝播遅延です。レジスタのクロック・ピンにおけるトータル・レイテンシ（またはクロック伝播遅延）は、クロック・パスにおけるソース・レイテンシとネットワーク・レイテンシの合計です。



`set_clock_latency` コマンドは、ソース・レイテンシのみをサポートしています。このコマンドを使用するときは、`-source` オプションを指定する必要があります。

`set_clock_latency` コマンドを使用して、デザイン内の任意のクロック・ポートまでのソース・レイテンシを指定します。例 7-18 に、`set_clock_latency` コマンドとオプションを示します。

例 7-18. set_clock_latency コマンド

```
set_clock_latency
-source
[-clock <clock_list>]
[-rise | -fall]
[-late | -early]
<delay>
<targets>
```

表 7-11 に、set_clock_latency コマンドのオプションを示します。

オプション	説明
-source	ソース・レイテンシを指定します。
-clock <clock list>	ターゲットに複数のクロックが割り当てられている場合に、どのクロックを使用するかを指定します。
-rise -fall	立ち上がり遅延または立ち下がり遅延を指定します。
-late -early	クロックへの最も早い到達時間、または最も遅い到達時間を指定します。
<delay>	遅延値を指定します。
<targets>	クロックが複数のクロックによって駆動されている場合に、クロックまたはクロック・ソースを指定します。

ネットワーク・レイテンシは、Quartus II TimeQuest タイミング・アナライザが自動的に計算します。したがって、set_clock_latency コマンドは、ソース・レイテンシのみを指定します。

クロック不確実性

set_clock_uncertainty コマンドは、クロックまたはクロック間転送のクロック不確実性またはスキューを指定します。不確実性をセットアップとホールドに別々に指定します。立ち上がりクロックと立ち下がりクロックの遷移を別々に指定します。Quartus II TimeQuest タイミング・アナライザは、適用可能な各パスのデータ要求時間からセットアップ不確実性を減算し、適用可能な各パスのデータ要求時間にホールド不確実性を加算します。

set_clock_uncertainty コマンドを使用して、クロック・ポートへのクロック不確実性を指定します。例 7-19 に、set_clock_uncertainty コマンドとオプションを示します。

例 7-19. set_clock_uncertainty コマンドとオプション

```
set_clock_uncertainty
[-rise_from <rise from clock> | -fall_from <fall from clock> |
-from <from clock>]
[-rise_to <rise to clock> | -fall_to <fall to clock> | -to <to clock>]
[-setup | -hold]
<value>
```

表 7-12 に、set_clock_uncertainty コマンドのオプションを示します。

表 7-12. set_clock_uncertainty コマンドのオプション	
オプション	説明
-from <from clock>	from clock を指定します。
-rise_from <rise from clock>	rise-from clock を指定します。
-fall_from <fall from clock>	fall-from clock を指定します。
-to <to clock>	to clock を指定します。
-rise_to <rise to clock>	rise-to clock を指定します。
-fall_to <fall to clock>	fall-to clock を指定します。
-setup -hold	setup または hold を指定します。
<value>	不確実性の値。

クロック不確実性の自動計算

derive_clock_uncertainty コマンドを使用して、インター・クロック、イントラ・クロック、および I/O インタフェースの不確実性を自動的に適用します。クロック間転送ごとに、セットアップ不確実性とホールド不確実性の両方が計算されます。例 7-20 に、derive_clock_uncertainty コマンドとオプションを示します。

例 7-20. derive_clock_uncertainty コマンド

```
derive_clock_uncertainty
[-overwrite]
[-add]
```

表 7-13 に、`derive_clock_uncertainty` コマンドのオプションを示します。

オプション	説明
<code>-overwrite</code>	前に実行したクロック不確実性アサインメントを上書きします。
<code>-add</code>	ユーザー定義のクロック不確実性アサインメントに、 <code>derive_clock_uncertainty</code> の結果を追加します。

Quartus II TimeQuest タイミング・アナライザは、デザインのクロック間転送にクロック不確実性を自動的に適用します。

`set_clock_uncertainty` コマンドによって、ソース・クロックと destinations・クロックのペアに適用したクロック不確実性の制約は、同じペアに対して `derive_clock_uncertainty` コマンドから派生したクロック不確実性よりも優先順位が高くなります。例えば、`set_clock_uncertainty` を `clka` と `clkb` の間に適用する場合、クロック転送の `derive_clock_uncertainty` 値はデフォルトでは無視されません。`set_clock_uncertainty` 制約は、`derive_clock_uncertainty` 制約よりも優先されます。

ただし、使用されるはずであったクロック不確実性は、情報目的でレポートされます。`-overwrite` コマンドを使用して前のクロック不確実性アサインメントを上書きするか、`remove_clock_uncertainty` コマンドを使用して手動で削除できます。また、`-add` オプションを使用して、`derive_clock_uncertainty` コマンドによって決定したクロック不確実性を、以前に定義したクロック不確実性値に追加することもできます。

次に示すのは、クロック不確実性が生じる可能性があるクロック間転送の種類です。これらは、`derive_clock_uncertainty` コマンドによって自動的にモデル化されます。

- イントラ・クロック
- インター・クロック
- I/O インタフェース

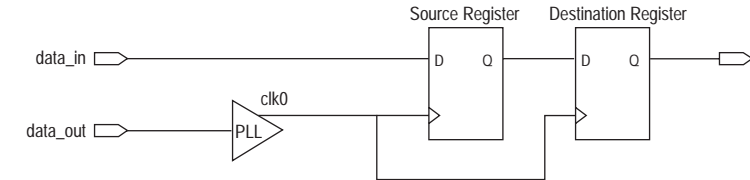


アルテラは、`derive_clock_uncertainty` コマンドを使用することを推奨しています。

イントラ・クロック転送

イントラ・クロック転送は、レジスタ間転送が FPGA のコアで発生し、ソース・クロックとデスティネーション・クロックが同じ PLL 出力ピンまたはクロック・ポートから供給される場合に発生します。図 7-28 にイントラ・クロック転送の一例を示します。

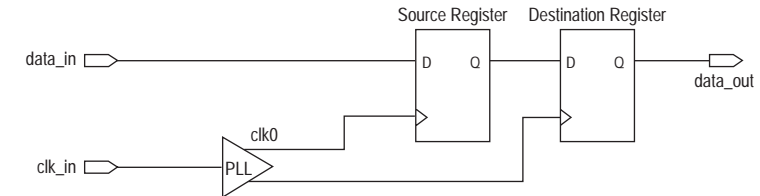
図 7-28. イントラ・クロック転送



インター・クロック転送

インター・クロック転送は、レジスタ間転送が FPGA のコアで発生し、ソース・クロックとデスティネーション・クロックがそれぞれ異なる PLL 出力ピンまたはクロック・ポートから供給される場合に発生します。図 7-29 に、インター・クロック転送の一例を示します。

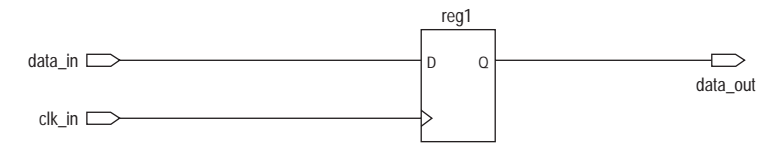
図 7-29. インター・クロック転送



I/O インタフェース・クロック転送

I/O インタフェース・クロック転送は、データが I/O ポートから FPGA のコアに（入力）、または FPGA のコアから I/O ポートに（出力）転送される場合に発生します。図 7-30 に、I/O インタフェース・クロック転送の一例を示します。

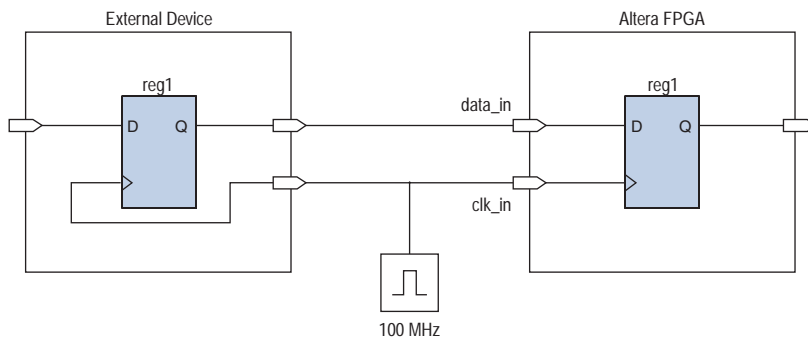
図 7-30. I/O インタフェース・クロック転送



I/O インタフェースの不確実性については、まず仮想クロックを作成し、次にその仮想クロックを参照する `set_input_delay` および `set_output_delay` コマンドを使用して、入力および出力ポートを制約する必要があります。 `set_input_delay` または `set_output_delay` コマンドがクロック・ポートまたは PLL 出力を参照するときに、 `derive_clock_uncertainty` コマンドによって、イントラ・クロック転送またはインター・クロック転送のクロック不確実性が I/O インタフェース・クロック転送に適用されないようにするために仮想クロックが必要です。 `set_input_delay` または `set_output_delay` コマンドで仮想クロックが参照されない場合は、 `derive_clock_uncertainty` コマンドで I/O インタフェースに対してイントラ・クロックまたはインター・クロック用の値を計算します。

I/O ポートをドライブする元のクロックと同じプロパティを持つ仮想クロックを作成します。例えば、図 7-31 に、クロック仕様を持つ標準的な入力 I/O インタフェースを示します。

図 7-31. I/O インタフェース仕様



例 7-21 に、図 7-31 に示す I/O インタフェースを制約する SDC コマンドを示します。

例 7-21. I/O インタフェースを制約する SDC コマンド

```
# クロック・ポートのベース・クロックを作成
create_clock -period 10 -name clk_in [get_ports clk_in]
# ソース・レジスタをドライブするベース・クロックと同じプロパティを持つ仮想クロックを作成
create_clock -period 10 -name virt_clk_in
# ベース・クロックではなく、
# 仮想クロックを参照する入力遅延を作成
# set_input_delay -clock clk_in <delay_value> [get_ports data_in] を使用しないこと
set_input_delay -clock virt_clk_in <delay_value> [get_ports data_in]
```

I/O 規格

Quartus II TimeQuest タイミング・アナライザは、デザイン内のポートを制約する SDC コマンドをサポートしています。これらの制約により、Quartus II TimeQuest タイミング・アナライザは、FPGA の内部タイミングだけでなく、外部デバイスのタイミングおよび外部ボードのタイミング・パラメータも含む、システム・スタティック・タイミング解析を実行できます。

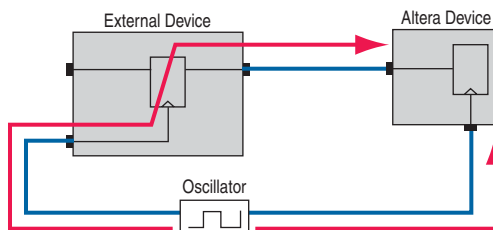
入力および出力遅延

外部デバイスまたは外部ボードのタイミング・パラメータを指定するには、入力および出力遅延制約を使用します。これらの制約を適用すると、Quartus II TimeQuest タイミング・アナライザは、システム全体でスタティック・タイミング解析を実行します。

Set Input Delay

set_input_delay 制約は、ポート（デバイス I/O）におけるクロックに対するデータ到達時間を指定します。図 7-32 に、入力遅延パスを示します。

図 7-32. Set Input Delay



set_input_delay コマンドを使用して、デザインのポートに入力遅延制約を指定します。例 7-22 に、set_input_delay コマンドとオプションを示します。

例 7-22. set_input_delay コマンド

```
set_input_delay
-clock <clock name>
[-clock_fall]
[-rise | -fall]
[-max | -min]
[-add_delay]
[-reference_pin <target>]
[-source_latency_included]
<delay value>
<targets>
```

表 7-14 に、set_input_delay コマンドのオプションを示します。

オプション	説明
-clock <clock name>	ソース・クロックを指定します。
-clock_fall	クロックの立ち下がりエッジに対する到達時間を指定します。
-rise -fall	ポートにおける立ち上がり遅延または立ち下がり遅延を指定します。
-max -min	最小または最大データ到達時間を指定します。
-add_delay	別の遅延を追加します。ただし、ポートに割り当てられている既存の遅延は置き換えません。
-reference_pin <target>	ソース・レイテンシとネットワーク・レイテンシを決定するデザインのピンまたはポートを指定します。これは、クロックが供給される出力ポートを基準にして入力遅延を指定するのに便利です。
-source_latency_included	入力遅延値にソース・レイテンシ遅延値が含まれることを指定します。したがって、クロックに割り当てられているソース・クロック・レイテンシは無視されます。
<delay value>	遅延値を指定します。
<targets>	デスティネーション・ポートまたはピンを指定します。



入力遅延値に `-max` または `-min` 値の一方しか指定していない場合は、警告メッセージが表示されます。入力最小遅延のデフォルト値は、入力最大遅延と同じです。いずれか一方しか指定していない場合は、入力最大遅延のデフォルト値が入力最小遅延と同じになります。同様に、遅延値に `-rise` または `-fall` 値の一方しか指定していない場合は警告メッセージが表示され、デフォルトの遅延値は、入力最小および入力最大遅延の場合と同じように設定されます。

最大値はセットアップ・チェックに、最小値はホールド・チェックに使用されます。

デフォルトでは、入力遅延のセット (`min/max`, `rise/fall`) は、`-clock`、`-clock_fall`、`-reference_pin` の組み合わせにのみ使用できます。別のクロックの同じポートで入力遅延 (`-clock_fall` または `-reference_pin`) を指定すると、`-add_delay` オプションを指定しない限り、以前設定された入力遅延はすべて削除されます。`-add_delay` オプションを指定するとワースト・ケース値が使用されます。

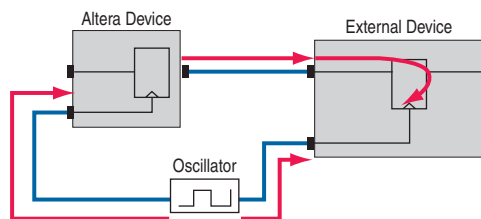
`-min` および `-max` オプションと同様、`-rise` および `-fall` オプションは相互排他的です。

Set Output Delay

`set_output_delay` コマンドは、ポート (デバイス・ピン) におけるクロックに対するデータ所要時間を指定します。

`set_output_delay` コマンドを使用して、デザインのポートに出力遅延制約を指定します。図 7-33 に出力遅延パスを示します。

図 7-33. 出力遅延



例 7-23 に、`set_output_delay` コマンドとオプションを示します。

例 7-23. `set_output_delay` コマンド

```
set_output_delay
-clock <clock name>
[-clock_fall]
[-rise | -fall]
[-max | -min]
[-add_delay]
[-reference_pin <target>]
<delay value>
<targets>
```

表 7-15 に、`set_output_delay` コマンドのオプションを示します。

オプション	説明
<code>-clock <clock name></code>	ソース・クロックを指定します。
<code>-clock_fall</code>	クロックの立ち下がりエッジに対する所要時間を指定します。
<code>-rise -fall</code>	ポートにおける立ち上がり遅延または立ち下がり遅延を指定します。
<code>-max -min</code>	最小または最大データ到達時間を指定します。
<code>-add_delay</code>	別の遅延を追加します。ただし、ポートに割り当てられている既存の遅延は置き換えません。
<code>-reference_pin <target></code>	ソース・レイテンシとネットワーク・レイテンシを決定するデザインのピンまたはポートを指定します。このオプションを使用して、クロックが供給される出力ポートを基準にして入力遅延を指定します。
<code>-source_latency_included</code>	入力遅延値にソース・レイテンシ遅延値が含まれることを指定します。したがって、それ以降、クロックに割り当てられているソース・クロック・レイテンシは無視されます。
<code><delay value></code>	遅延値を指定します。
<code><targets></code>	デスティネーション・ポートまたはピンを指定します。



出力遅延値に `-max` または `-min` 値の一方しか指定していない場合は、警告メッセージが表示されます。出力最小遅延のデフォルト値は、出力最大遅延です。いずれか一方しか指定していない場合は、出力最大遅延のデフォルト値が出力最小遅延です。

最大値はセットアップ・チェックに、最小値はホールド・チェックに使用されます。

デフォルトでは、出力遅延のセット（min/max、rise/fall）は、1 つのクロック、-clock_fall、ポートの組み合わせにのみ使用できます。別のクロックまたは-clock_fallの同じポートで出力遅延を指定すると、-add_delay オプションを指定しない限り、以前設定された出力遅延はすべて削除されます。-add_delay オプションを指定するとワースト・ケース値が使用されます。

-minおよび-maxオプションと同様、-riseおよび-fallオプションは相互排他的です。

タイミング例外

タイミング例外は、Quartus II TimeQuest タイミング・アナライザが実行するデフォルトの解析を修正します。この項では、以下の使用可能なタイミング例外について説明します。

- 7-58 ページの「フォルス・パス」
- 7-59 ページの「最小遅延」
- 7-60 ページの「最大遅延」
- 7-62 ページの「マルチサイクル・パス」

優先順位

タイミング例外の間でノード名が競合する場合、以下の優先順位が適用されます。

1. フォルス・パス
2. 最小遅延および最大遅延
3. マルチサイクル・パス

フォルス・パスのタイミング例外が最優先されます。各カテゴリ内では、個々のノードに対するアサインメントがクロックに対するアサインメントよりも優先されます。最後に、さらに競合が発生した場合、残りの優先順位は順序によって決まり、最後のアサインメントが以前のアサインメントを上書き（または一部のみ上書き）します。

フォルス・パス

フォルス・パスはタイミング解析で無視できるパスです。

set_false_path コマンドを使用して、デザインのフォルス・パスを指定します。例 7-24 に、set_false_path コマンドとオプションを示します。

例 7-24. set_false_path コマンド

```
set_false_path
[-fall_from <clocks> | -rise_from <clocks> | -from <names>]
[-fall_to <clocks> | -rise_to <clocks> | -to <names>]
[-hold]
[-setup]
[-through <names>]
<delay>
```

表 7-16 に、set_false_path コマンドのオプションを示します。

オプション	説明
-fall_from <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。フォルス・パスが <clocks> からの立ち下がりで開始することを指定します。
-fall_to <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。フォルス・パスが <clocks> への立ち下がりで終了することを指定します。
-from <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。フォルス・パスが <names> で始まることを指定します。
-hold	フォルス・パスがホールド解析時にのみ有効であることを指定します。
-rise_from <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。フォルス・パスが <clocks> からの立ち上がりで開始することを指定します。
-rise_to <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。フォルス・パスが <clocks> への立ち上がりで終了することを指定します。
-setup	フォルス・パスがセットアップ解析時にのみ有効であることを指定します。
-through <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。フォルス・パスが <names> を通過することを指定します。
-to <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。フォルス・パスが <names> で終了することを指定します。
<delay>	遅延値を指定します。

オブジェクトがタイミング・ノードの場合、フォルス・パスは2つのノード間のパスにのみ適用されます。オブジェクトがクロックの場合、ソース・ノード (-from) またはデステイネーション・ノード (-to) がクロックによって駆動されるすべてのパスに適用されます。

最小遅延

set_min_delay コマンドを使用して、特定のパスの絶対最小遅延を指定します。以下に、set_min_delay コマンドとオプションを示します。

例 7-25. set_min_delay コマンド

```
set_min_delay
[-fall_from <clocks> | -rise_from <clocks> | -from <names>]
[-fall_to <clocks> | -rise_to <clocks> | -to <names>]
[-through <names>]
<delay>
```

表 7-17 に、set_min_delay コマンドのオプションを示します。

表 7-17. set_min_delay コマンドのオプション	
オプション	説明
-fall_from <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。最小遅延が <clocks> の立ち下がりエッジで開始することを指定します。
-fall_to <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。最小遅延が <clocks> の立ち下がりエッジで終了することを指定します。
-from <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの始点になります。
-rise_from <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<clocks> の立ち上がりエッジの最小遅延を指定します。
-rise_to <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<clocks> の立ち上がりエッジの最小遅延を指定します。
-through <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの通過点になります。
-to <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの終点になります。
<delay>	遅延値を指定します。

ソース・ノードまたはデスティネーション・ノードがクロックされる場合は、クロック・パスを考慮して、データ・パスにある程度の遅延を許容します。ソース・ノードまたはデスティネーション・ノードに入力遅延または出力遅延がある場合、その遅延も最小遅延チェックに含まれません。

オブジェクトがタイミング・ノードの場合、最小遅延は2つのノード間のパスにのみ適用されます。オブジェクトがクロックの場合、最小遅延はソース・ノード (-from) またはデスティネーション・ノード (-to) がクロックによって駆動されるすべてのパスに適用されます。

set_min_delay コマンドは、set_output_delay 制約を使用しない出力ポートを除いて適用できます。この場合、セットアップ・サマリとホールド・サマリがこれらのパスのスラックをレポートします。出力ポートに関連付けられているクロックがないため、これらのパスにクロックはレポートされず、**Clock** カラムは空白です。この場合、これらのパスのタイミングはレポートできません。



set_min_delay コマンドで、出力パスにクロック・フィルタを使用してタイミグをレポートするには、値 0 の出力ポートに set_output_delay コマンドを使用できます。set_output_delay コマンドでは、デザインからの既存のクロックまたは仮想クロックをクロック基準として使用できます。

最大遅延

set_max_delay コマンドを使用して、特定のパスの絶対最大遅延を指定します。例 7-26 に、set_max_delay コマンドとオプションを示します。

例 7-26. set_max_delay コマンド

```
set_max_delay
[-fall_from <clocks> | -rise_from <clocks> | -from <names>]
[-fall_to <clocks> | -rise_to <clocks> | -to <names>]
[-through <names>]
<delay>
```

表 7-18 に、set_max_delay コマンドのオプションを示します。

表 7-18. set_max_delay コマンドのオプション	
オプション	説明
-fall_from <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。最大遅延が <clocks> の立ち下がりエッジで開始することを指定します。
-fall_to <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。最大遅延が <clocks> の立ち下がりエッジで終了することを指定します。
-from <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの始点になります。
-rise_from <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<clocks> の立ち上がりエッジの最大遅延を指定します。
-rise_to <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<clocks> の立ち上がりエッジの最大遅延を指定します。
-through <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの通過点になります。
-to <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの終点になります。
<delay>	遅延値を指定します。

ソース・ノードまたはデスティネーション・ノードがクロックされる場合は、クロック・パスを考慮して、データ・パスにある程度の遅延を許容します。ソース・ノードまたはデスティネーション・ノードに入力遅延または出力遅延がある場合、その遅延も最大遅延チェックに含まれます。

オブジェクトがタイミング・ノードの場合、最大遅延は2つのノード間のパスにのみ適用されます。オブジェクトがクロックの場合は、ソース・ノード (-from) またはデスティネーション・ノード (-to) がクロックによって駆動されるすべてのパスに適用されます。

set_max_delay コマンドは、set_output_delay 制約を使用しない出力ポートを除いて適用できます。この場合、セットアップ・サマリとホールド・サマリがこれらのパスのスラックをレポートします。出力ポートに関連付けられているクロックがないため、これらのパスにクロックはレポートされず、**Clock** カラムは空白です。この場合、これらのパスのタイミングはレポートできません。



set_max_delay コマンドで、出力パスにクロック・フィルタを使用してタイミングをレポートするには、値 0 の出力ポートに set_output_delay コマンドを使用できます。set_output_delay コマンドでは、デザインからの既存のクロックまたは仮想クロックをクロック基準として使用できます。

マルチサイクル・パス

デフォルトでは、Quartus II TimeQuest タイミング・アナライザは、シングル・サイクル解析を使用します。パスの解析では、セットアップ・ローンチおよびラッチ・エッジ時間は、それぞれの波形で最も近い 2 つのアクティブ・エッジを見つけることによって決定されます。ホールド解析の場合、タイミング・アナライザは 2 つのタイミング条件を基準にしてパスを解析し、ワースト・ケース・セットアップ関係だけでなく、可能なすべてのセットアップ関係を探します。したがって、ホールド・ローンチ時間およびラッチ時間は、セットアップ・ローンチ・エッジおよびラッチ・エッジとは完全に無関係になる場合があります。

マルチサイクル制約は、ソース (-start) またはデステイネーション (-end) クロックに基づいて、指定されたクロック・サイクル数だけ、セットアップまたはホールド関係を緩和します。2 のエンド・マルチサイクル制約は、ワースト・ケースのセットアップ・ラッチ・エッジを 1 デステイネーション・クロック周期だけ拡張します。

ホールド・マルチサイクル制約は、デフォルトのホールド位置 (デフォルト値は 0) に基づきます。1 のエンド・マルチサイクル制約は、デフォルトのホールド・ラッチ・エッジから 1 デステイネーション・クロック周期を効果的に減算します。

set_multicycle_path コマンドを使用して、デザインのマルチサイクル制約を指定します。例 7-27 に、set_multicycle_path コマンドとオプションを示します。

例 7-27. set_multicycle_path コマンド

```
set_multicycle_path
[-end]
[-fall_from <clocks> | -rise_from <clocks> | -from <names>]
[-fall_to <clocks> | -rise_to <clocks> | -to <names>]
[-hold]
[-setup]
[-start]
[-through <names>]
<path multiplier>
```

表 7-19 に、set_multicycle_path コマンドのオプションを示します。

表 7-19. set_multicycle_path コマンドのオプション	
オプション	説明
-fall_from <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。マルチサイクルが <clocks> の立ち下がりエッジで開始することを指定します。
-fall_to <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。マルチサイクルが <clocks> の立ち下がりエッジで終了することを指定します。
-from <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの始点になります。
-hold -setup	適用するマルチサイクルの種類を指定します。
-rise_from <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。マルチサイクルを <clocks> の立ち上がりエッジで指定します。
-rise_to <clocks>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。マルチサイクルが <clocks> の立ち上がりエッジで終了することを指定します。
-start -end	開始または終了クロックが、マルチサイクルのソースまたはデスティネーションとして機能するかどうかを指定します。
-through <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。マルチサイクルが <names> を通過することを指定します。
-to <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの終点になります。
<path multiplier>	マルチサイクルの乗算器の値を指定します。

オブジェクトがタイミング・ノードの場合、マルチサイクル制約は、2つのノード間のパスにのみ適用されます。オブジェクトがクロックの場合は、ソース・ノード (-from) またはデスティネーション・ノード (-to) がクロックによって駆動されるすべてのパスに適用されます。

アプリケーション例

この項では、set_multicycle_path コマンドの具体的な例を示します。

図 7-34 に、レジスタ間パスを示します。ここで、ソース・クロック src_clk の周期は 10 ns、デスティネーション・クロック dst_clk の周期は 5 ns です。

図 7-34. レジスタ間パス

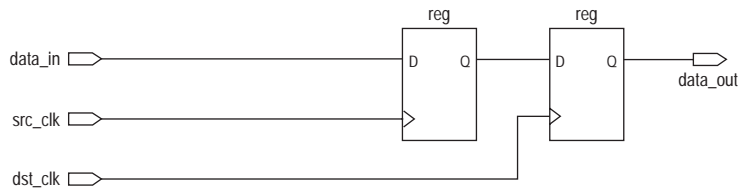
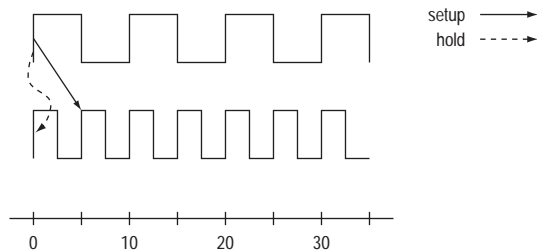


図 7-35 に、ソース・クロックおよびデスティネーション・クロックのタイミング図と、デフォルトのセットアップ関係およびホールド関係を示します。デフォルトのセットアップ関係は 5 ns、デフォルトのホールド関係は 0 ns です。

図 7-35. デフォルトのセットアップおよびホールドのタイミング図



デフォルトのセットアップおよびホールド関係は、システム要件を満たすように `set_multicycle_path` コマンドで修正することができます。

表 7-20 に、Quartus II TimeQuest タイミング・アナライザがセットアップ関係またはホールド関係を決定するのに使用する、ローンチ・エッジ時間またはラッチ・エッジ時間を修正するためのコマンドを示します。

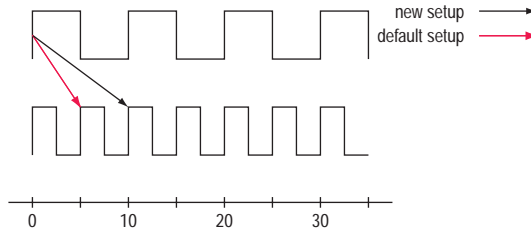
表 7-20. エッジ時間修正コマンド	
コマンド	修正の説明
<code>set_multicycle_path -setup -end</code>	セットアップ関係のラッチ・エッジ時間
<code>set_multicycle_path -setup -start</code>	セットアップ関係のローンチ・エッジ時間
<code>set_multicycle_path -hold -end</code>	ホールド関係のラッチ・エッジ時間
<code>set_multicycle_path -hold -start</code>	ホールド関係のローンチ・エッジ時間

図 7-36 に、セットアップ・ラッチ・エッジを修正するコマンドと、修正後のタイミング図を示します。このコマンドにより、ラッチ・エッジ時間をデフォルトの 5 ns から 10 ns に移動します。

図 7-36. 修正後のセットアップ図

#2 番目のエッジごとにラッチ

```
set_multicycle_path -from [get_clocks src_clk] -to [get_clocks dst_clk] -setup -end 2
```



制約と例外の削除

Quartus II TimeQuest タイミング・アナライザを対話的に使用する場合は通常、制約または例外を削除する必要があります。古くなったり、誤って入力されている制約や例外は、Quartus II TimeQuest タイミング・アナライザで容易に削除できます。

表 7-21 に、デザインから制約と例外を削除できるコマンドを示します。

コマンド	説明
remove_clock [-all] [<clock list>]	<clock list> で指定される作成済みクロックを削除します。-all オプションは、宣言されているすべてのクロックを削除します。
remove_clock_groups -all	以前に作成されたすべてのクロック・グループを削除します。特定のクロック・グループは削除できません。
remove_clock_latency -source <targets>	<targets> で指定されるクロックからクロック・レイテンシ制約を削除します。
remove_clock_uncertainty -from <from clock> -to <to clock>	<from clock> から <to clock> までのクロック不確実性制約を削除します。
remove_input_delay <targets>	<targets> から入力遅延制約を削除します。
remove_output_delay <targets>	<targets> から出力遅延制約を削除します。
reset_design	デザインのすべての制約と例外を削除します。

タイミング・ レポート

Quartus II TimeQuest タイミング・アナライザは、スタティック・タイミング解析結果をリアルタイムでレポートします。レポートは要求されたときのみ生成されます。必須以外のフィールドを除外して、特定のタイミング情報を表示するレポートをカスタマイズできます。

この項では、Quartus II TimeQuest タイミング・アナライザでサポートされる各種レポート生成コマンドについて説明します。

report_timing

report_timing コマンドを使用して、セットアップ、ホールド、リカバリ、またはリムーバル・レポートを生成します。例 7-28 に、report_timing コマンドとオプションを示します。

例 7-28. report_timing コマンド

```
report_timing
[-append]
[-detail <summary|path_only|path_and_clock|full_path>]
[-fall_from_clock <names>]
[-fall_to_clock <names>]
[-false_path]
[-file <name>]
[-from <names>]
[-from_clock <names>]
[-hold]
[-less_than_slack <slack limit>]
[-npaths <number>]
[-nworst <number>]
[-pairs_only]
[-panel_name <name>]
[-recovery]
[-removal]
[-rise_from_clock <names>]
[-rise_to_clock <names>]
[-setup]
[-show_routing]
[-stdout]
[-through <names>]
[-to <names>]
[-to_clock <names>]
```

表 7-22 に、report_timing コマンドのオプションを示します。

表 7-22. report_timing コマンドのオプション (1 / 2)	
オプション	説明
-append	現在のレポートを -file オプションで指定されるファイルに追加するよう指定します。
-detail <summary path_only path_and_clock full_path>	クロック・パスの詳細をレポートするかどうかを指定します。 Path Only: クロック・ネットワーク遅延がまとめられます。 Summary: 個々のパスをリストします。 Path and Clock: クロック・ネットワーク遅延が詳細に示されます。 Full Path: より多くのクロック・ネットワークの詳細、特に生成クロックについて示します。
-fall_from_clock <names>	解析するソース・レジスタからの <names> の立ち下がりエッジを指定します。オプション from_clock、fall_from_clock、および rise_from_clock は相互排他的です。
-fall_to_clock <names>	解析するデスティネーション・レジスタへの <names> の立ち下がりエッジを指定します。オプション to_clock、fall_to_clock、および rise_to_clock は相互排他的です。
-false_path	フォルス・パス・アサインメントによってカットされるパスのみをレポートします。
-file <name>	現在のレポートをファイル <name> に書き込むことを示します。
-hold	クロック・ホールド解析を指定します。
-less_than_slack <slack limit>	レポートするパスを <slack limit> 値に制限します。
-npaths <number>	レポートするパス数を指定します。
-nworst <number>	エンドポイントあたりのパス数を制限します。
-panel_name <names>	Reports ペインにパネル名を指定します。
-panel_name <names>	パネルに結果を送信し、新しいパネルの名前を指定します。
-pairs_only	これを設定すると、同じ始点と終点を持つパスは同等なパスとみなされ、それぞれの固有の組み合わせのワースト・ケース・パスのみ表示されます。
-recovery	リカバリ解析を指定します。
-removal	リムーバル解析を指定します。
-rise_from_clock <names>	解析するソース・レジスタからの <names> の立ち上がりエッジを指定します。オプション from_clock、fall_from_clock、および rise_from_clock は相互排他的です。
-rise_to_clock <names>	解析するデスティネーション・レジスタへの <names> の立ち上がりエッジを指定します。オプション to_clock、fall_to_clock、および rise_to_clock は相互排他的です。
-setup	クロック・セットアップ解析を指定します。
-show_routing	パスの詳細な配線を表示するオプション。

表 7-22. report_timing コマンドのオプション (2 / 2)

オプション	説明
-stdout	レポートを stdout に送信することを示します。
-through <names>	解析用の通過ノードを指定します。
-to <names>	解析用のノードを指定します。
-to_clock <names>	解析用のデスティネーション・クロックを指定します。


例 7-29 に示すサンプル・レポートは、次のコマンドを入力した結果です。

```
report_timing -from_clock clk_async -to_clock clk_async -setup -npaths 1 ↓
```

例 7-29. report_timing レポートの例

```
Info:
=====
Info: To Node      :dst_reg
Info: From Node    :src_reg
Info: Latch Clock  :clk_async
Info: Launch Clock :clk_async
Info:
Info: Data Arrival Path:
Info:
Info: Total (ns)  Incr (ns)      Type  Node
Info: =====  =====  ==  =====
Info: 0.000      0.000           launch edge time
Info: 2.237      2.237  R           clock network delay
Info: 2.410      0.173    uTco    src_reg
Info: 2.410      0.000  RR    CELL  src_reg|regout
Info: 3.407      0.997  RR    IC    dataout|datain
Info: 3.561      0.154  RR    CELL  dst_reg
Info:
Info: Data Required Path:
Info:
Info: Total (ns)  Incr (ns)      Type  Node
Info: =====  =====  ==  =====
Info: 10.000     10.000           latch edge time
Info: 11.958     1.958  R           clock network delay
Info: 11.610     -0.348    uTsu    dst_reg
Info:
Info: Data Arrival Time      :      3.561
Info: Data Required Time    :      11.610
Info: Slack                  :      8.049
Info: =====
```

report_timing コマンドは、セットアップ、ホールド、リカバリ、またはリムーバルのうち、指定された解析タイプについてのレポートを生成します。各カラムについては、表 7-23 で説明します。

 レポート・パネルが作成される時のみ、すべてのカラムが表示されます。report_timing 出力がファイルまたはコンソールに転送される場合、Total、Incr、RF、Type、および Node カラムのみ表示されます。

カラム名	説明
Total	時間遅延の累積値を示します。
Incr	遅延の増分量を示します。
RF	エレメントの入力および出力遷移を示します。これは、R、F、RR、RF、FR、FF のいずれかです。
Type	ノード・タイプを示します。各種ノード・タイプの説明は、表 7-24 を参照してください。
Fanout	エレメントのファンアウト数を示します。
Location	FPGA でのエレメントの位置を示します。
Element	エレメントの名前を示します。

表 7-24 に、report_timing レポートに表示されるノード・タイプの説明を示します。

タイプ名	説明
CELL	エレメントが、FPGA 内のレジスタまたはエレメントの組み合わせのいずれであるかを示します。CELL は、ALM のレジスタ、メモリ・ブロック、DSP ブロック、または I/O ブロックのいずれかです。
COMP	PLL クロック・ネットワーク補正遅延を示します。
IC	エレメントがインタコネクタ遅延であることを示します。
μt_{CO}	エレメントのマイクロ clock-to-out 時間を示します。
μt_{SU}	エレメントのマイクロ・セットアップ時間を示します。
μt_{H}	エレメントのマイクロ・ホールド時間を示します。
i_{EXT}	エレメントの外部入力遅延時間を示します。
o_{EXT}	エレメントの外部出力遅延時間を示します。

report_metastability

report_metastability ファンクションを使用して、デザインにおける非同期転送の堅牢性を見積もり、検出されるすべての同期レジスタ・チェーンの MTBF を詳細に示すレポートを生成します。

例 7-30. report_metastability コマンド

```
report_metastability
[-append]
[-file <name>]
[-panel_name <name>]
[-stdout]
```

表 7-25 に、report_metastability コマンドのオプションを示します。

オプション	説明
-append	現在のレポートを -file で指定されるファイルに追加するよう指定します。
-file <name>	結果をファイルに送信します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。
-stdout	レポートを stdout に送信するよう指示します。

report_metastability ファンクションは、デザイン内の同期レジスタ・チェーンのリストを生成し、各チェーンの MTBF を見積もります。デザインの MTBF は、デザインの全体的な堅牢性を見積もりであり、すべての同期チェーンの MTBF の結果から計算されます。

この MTBF の見積もりでは、選択されたデバイス・スピード・グレードの標準シリコン特性として定義される標準的な条件や、公称電圧および温度（25℃）を仮定しています。

MTBF の見積もりでは、同期されるデータはソース・クロック周波数の 12.5% のトグル・レートで切り替わるものと仮定しています。つまり、これらの見積もりは、到達するデータが 8 ソース・クロック・サイクルごとに 1 回切り替わるものと仮定して行われます。複数のクロックが適用される場合は、最高周波数が使用されます。ソース・クロックを決定できない場合、データ・レートは同期クロック周波数の 12.5% とみなされます。

デザインは完全に制約され、TimeQuest タイミング・アナライザで使用可能な機能のタイミングを渡す必要があります。入力ポートが同期レジスタとして動作していないレジスタにファンアウトする場合、その入力ポートに `set_input_delay` 制約を適用することが重要です。この制約を適用しないと、TimeQuest タイミング・アナライザはこの入力ポートを非同期入力とみなすため、これらのレジスタは同期レジスタとしてレポートされます。同様に、フォルス・パスの終端にあるレジスタも同期レジスタとみなされます。

同期入力ポートを `set_max_delay` 制約で制約する場合、これによってシンクロナイザの識別が妨げられることはありません。`set_max_delay` 制約は、入力ポートをクロック・ドメインに関連付けないため、この入力ポートは非同期とみなされます。`set_max_delay` 制約を使用する代わりに、`set_input_delay -max` 制約を使用します。

例えば、セットアップ要件を次のように指定するのではなく、

```
set_max_delay < $t_{SU}$  requirement>
```

次の構文を使用します。

```
set_input_delay -max <latch-launch- $t_{SU}$  requirement>
```

設定

TimeQuest タイミング・アナライザは、デザインのメタスタビリティを解析できるため、**Metastability Analysis** と **Synchronization Register Chain Length** の両方のオプションを設定する必要があります。


Metastability Analysis

このオプションは、TimeQuest タイミング・アナライザおよびフィックがレジスタを同期レジスタ・チェーンの一部として識別する方法をコントロールします。**Metastability Analysis** オプションの推奨設定は **AUTO** です。

Metastability Analysis オプションの各種設定の定義を表 7-26 に示します。

設定	説明
ON	<ul style="list-style-type: none"> 必要に応じて、ロジックを横断して同期レジスタを識別します。 長さに関係なく、検出したすべての同期レジスタをレポートします。
AUTO	<ul style="list-style-type: none"> ロジックを含まない同期チェーンを識別します。 1 より長い同期レジスタ・チェーンのみをレポートします。
OFF	<ul style="list-style-type: none"> 同期レジスタは識別されません。

Metastability Analysis オプションを調整するには、Quartus II ソフトウェアで、Settings メニューの **Timing Analysis Settings** の下にある TimeQuest Timing Analyzer ページをポイントします。Metastability Analysis の横のドロップダウン・メニューから希望の設定を選択します。

 **Metastability Analysis** の設定を変更した場合は、フィッタを再実行して新しい設定を反映させる必要があります。

Assignment Editor はグローバル・アサインメントをサポートするほか、ノードおよびエンティティ・レベルでの **Metastability Analysis** オプションをサポートしています。このオプションを使用すると、ノードまたはエンティティを指定し、デザインの特定期間に対するメタスタビリティ解析をオフにできます。一部の同期チェーンが誤って識別され、それらをレポートから削除する場合は、これらのパスの解析をオフにできます。これを行うには、Assignment Editor で、これらの同期チェーンの最初のレジスタに対して **Analyze Metastability** を **OFF** に設定します。このオプションをオフにすると、これらの同期チェーンに対してメタスタビリティ解析は実行されません。

逆に、デザインで検出されなかった同期チェーンがある場合は、このチェーンの最初のレジスタに対して **Analyze Metastability** オプションをオンにすることができ、同期チェーンとしての基準を満たしていればレポートされます。これは通常、同期チェーンのレジスタ間にロジックが存在する場合に発生することがあります。メタスタビリティ解析のデフォルト・モードでは、これらの構造はシンクロナイザとみなされません。

Synchronization Register Chain Length

Synchronization Register Chain Length オプションを使用すると、同期チェーンに含まれるレジスタに対して、Quartus II ソフトウェア最適化アルゴリズムは実行されません。例えば、**Synchronization Register Chain Length** を 2 に設定すると、レジスタ重複またはロジック・リタイミングなどの最適化は、識別されたすべての同期チェーン内の最初の 2 個のレジスタに対しては実行されません。**Synchronization Register Chain Length** オプションのデフォルト設定は 2 です。

Synchronization Register Chain Length オプションを変更するには、Settings メニューで **Analysis & Synthesis** をポイントします。**More Settings** をクリックします。**More Settings** ダイアログ・ボックスが表示されます。このダイアログ・ボックスから、**Synchronization Register Chain Length** を設定できます。

Assignment Editor でノードまたはエンティティの **Synchronization Register Chain Length** を設定することも可能です。この値を同期チェーンの先頭に設定して、このチェーン内のレジスタのみ指定した値まで保護することができます。これが役立つのは、特定の同期チェーンを 3 番目のレジスタまで保護するが、グローバル・デフォルト保護レベルは 2 にする場合です。

アルテラでは、このオプションをデザイン内の同期チェーンの最大長に設定して、すべての同期レジスタを保持することを推奨しています。

report_clock_transfers

report_clock_transfers コマンドを使用して、デザイン内のすべてのクロック間転送を詳細に示すレポートを生成します。クロック間転送は、2 つの異なるクロックによって駆動される 2 個のレジスタ間にパスが存在する場合にレポートされます。また、デステイネーション数やソース数などの情報もレポートされます。

report_clock_transfers コマンドを使用して、セットアップ、ホールド、リカバリ、またはリムーバル・レポートを生成します。

例 7-31 に、`report_clock_transfers` コマンドとオプションを示します。

例 7-31. report_clock_transfers コマンド

```
report_clock_transfers
[-append]
[-file <name>]
[-hold]
[-setup]
[-stdout]
[-recovery]
[-removal]
[-panel_name <name>]
```

表 7-27 に、`report_clock_transfers` コマンドのオプションを示します。

オプション	説明
-append	現在のレポートを -file で指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むことを指示します。
-hold	ホールド解析用に、クロック転送に関するサマリを作成します。
-setup	セットアップ解析用に、クロック転送に関するサマリを作成します。
-stdout	レポートを stdout に送信するよう指示します。
-recovery	リカバリ解析用に、クロック転送に関するサマリを作成します。
-removal	リムーバル解析用に、クロック転送に関するサマリを作成します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

report_clocks

`report_clocks` コマンドを使用して、デザイン内のすべてのクロックを詳細に示すレポートを生成します。このレポートには、デザイン内のすべてのクロックのタイプ、周期、波形（立ち上がりと立ち下がり）、ターゲットなどの情報が含まれています。

例 7-32 に、report_clocks コマンドとオプションを示します。

例 7-32. report_clocks コマンド

```
report_clocks
[-append]
[-desc]
[-file <name>]
[-stdout]
[-panel_name <name>]
```

表 7-28 に、report_clocks コマンドのオプションを示します。

表 7-28. report_clocks コマンドのオプション	
オプション	説明
-append	現在のレポートを -file で指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むことを示します。
-desc	クロック名を降順でソートするよう指定します。デフォルトは昇順です。
-stdout	レポートを stdout に送信するよう指示します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

report_min_pulse_width

report_min_pulse_width コマンドは、クロックの High または Low パルスが、クロック信号の実際の変化として認識されるのに十分な時間持続するかチェックします。最小パルス幅チェックが失敗した場合、レジスタがクロック遷移を認識しない可能性があります。report_min_pulse_width コマンドを使用して、デザイン内のすべてのクロックの最小パルス幅を詳細に示すレポートを生成します。このレポートには、デザイン内のすべてのクロックの High および Low パルスに関する情報が含まれています。

report_min_pulse_width コマンドは、RAM および DSP の最小周期チェックや、入力および出力クロック・ポートの I/O エッジ・レート制限もレポートします。出力ポートの場合、ポートにはクロック（または生成クロック）が割り当てられているか、入力 / 出力遅延のために -reference_pin として使用される必要があります。

report_min_pulse_width コマンドは、I/O エッジ・レート制限をチェックしますが、出力クロック・ポートのチェックは必ずしも実行しません。report_min_pulse_width コマンドで、出力クロック・ポートの I/O エッジ・レート制限をチェックする場合、出力クロック・ポートは以下のいずれかのカテゴリに分類する必要があります。

- クロックまたは生成クロック制約が割り当てられている。

または

- 入力または出力遅延制約のために -reference_pin を使用する。

デザインの各レジスタは、レジスタをクロックする 1 クロックあたり 2 回レポートされます。1 回目は High パルスの場合、2 回目は Low パルスの場合です。例 7-33 に、report_min_pulse_width コマンドとオプションを示します。

例 7-33. report_min_pulse_width コマンド

```
report_min_pulse_width
[-append]
[-file <name>]
[-nworst <number>]
[-stdout]
[<targets>]
[-panel_name <name>]
```

表 7-29 に、report_min_pulse_width コマンドのオプションを示します。

オプション	説明
-append	出力がファイルに送信される場合、このオプションは結果をそのファイルに追加します。そうでない場合、ファイルは上書きされます。
-file <name>	結果をファイルに送信します。
-nworst <number>	レポートするパルス幅チェック数を指定します。デフォルトは 1 です。
-stdout	メッセージを経由して出力を stdout にリダイレクトします。このオプションは、ファイルなどの別の出力フォーマットが選択され、またメッセージを受信する場合のみです。
-targets	レジスタを指定します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

report_net_timing

report_net_timing コマンドを使用して、デザイン内のネットについての遅延およびファンアウト情報を詳細に示すレポートを生成します。ネットはセルの出力ピンに対応します。

例 7-34 に、report_net_timing コマンドとオプションを示します。

例 7-34. report_net_timing コマンド

```
report_net_timing
[-append]
[-file <name>]
[-nworst_delay <number>]
[-nworst_fanout <number>]
[-stdout]
[-panel_name <name>]
```

表 7-30 に、report_net_timing コマンドのオプションを示します。

オプション	説明
-append	現在のレポートを-file オプションで指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むよう指示します。
-nworst_delay <number>	<number> のワースト・ネット遅延をレポートするよう指定します。
-nworst_fanout <number>	<number> のワースト・ネット・ファンアウトをレポートするよう指定します。
-stdout	レポートを stdout に送信するよう指示します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

report_sdc

report_sdc コマンドを使用して、プロジェクト内のすべての SDC 制約のレポートを生成します。

例 7-35 に、report_sdc コマンドとオプションを示します。

例 7-35. report_sdc コマンド

```
report_sdc
[-ignored]
[-append]
[-file]
[-stdout]
[-panel_name <name>]
```

表 7-31 に、report_sdc コマンドのオプションを示します。

オプション	説明
-ignored	無視されたアサインメントをレポートします。
-append	現在のレポートを -file で指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むよう指示します。
-stdout	レポートを stdout に送信するよう指示します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

report_ucp

report_ucp コマンドを使用して、デザイン内の制約されていないすべてのパスのレポートを生成します。

例 7-36 に、report_ucp コマンドとオプションを示します。

例 7-36. report_ucp コマンド

```
report_ucp
[-append]
[-file <name>]
[-hold]
[-setup]
[-stdout]
[-summary]
[-panel_name <name>]
```

表 7-32 に、report_ucp コマンドのオプションを示します。

オプション	説明
-append	現在のレポートを -file で指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むよう指示します。
-hold	制約されていないすべてのホールド・パスをレポートします。
-setup	制約されていないすべてのセットアップ・パスをレポートします。
-stdout	レポートを stdout に送信するよう指示します。
-summary	サマリ・パネルのみを生成します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

表 7-33 に、Quartus II TimeQuest タイミング・アナライザで使用可能なすべてのレポート・コマンドをまとめます。

Task ペインのレポート	Tcl コマンド	説明
Report Setup Summary	create_timing_summary -setup	すべての定義されているクロックのクロック・セットアップ・サマリを生成します。
Report Hold Summary	create_timing_summary -hold	すべての定義されているクロックのクロック・ホールド・サマリを生成します。
Report Recovery Summary	create_timing_summary -recovery	すべての定義されているクロックのクロック・リカバリ・サマリを生成します。
Report Removal Summary	create_timing_summary -removal	すべての定義されているクロックのクロック・リムーバル・サマリを生成します。
Report Clocks	report_clocks	すべての定義されているクロックのクロック・サマリを生成します。
Report Clock Transfers	report_clock_transfers	デザインのすべてのクロック間転送のクロック転送サマリを生成します。
Report SDC	report_sdc	すべての SDC ファイル・コマンドによる読み出しのサマリを生成します。
Report Unconstrained Paths	report_ucp	デザインの制約されていないすべてのパスのサマリを生成します。
Report Timing	report_timing	デザインの特定のパスの詳細サマリを生成します。
Report Net Timing	report_net_timing	デザインの特定のネットの詳細サマリを生成します。

Task ペインのレポート	Tcl コマンド	説明
Report Minimum Pulse Width	report_min_pulse_width	デザインの特定のレジスタの詳細サマリを生成します。
Create Slack Histogram	create_slack_histogram	デザインの特定のクロックの詳細ヒストグラムを生成します。

report_path

report_path コマンドを使用して、最長遅延パスと対応する遅延値をレポートします。

例 7-37 に、report_path コマンドとオプションを示します。

例 7-37. report_path コマンド

```
report_path
[-append]
[-file <name>]
[-from <names>]
[-npaths <number>]
[-stdout]
[-through]
[-to <names>]
[-panel_name <name>]
```

表 7-34 に、report_path コマンドのオプションを示します。

オプション	説明
-append	現在のレポートを -file オプションで指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むよう指示します。
-from <names>	解析用のソース・ノードを指定します。
-npaths <number>	レポートするパス数を指定します。
-stdout	レポートを stdout に送信するよう指示します。
-through <name>	解析用の通過ノードを指定します。
-to <names>	解析用のデスティネーション・ノードを指定します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

report_bottleneck

report_bottleneckコマンドを使用して、ワースト1,000のセットアップ・パスを対象に、各ノードを通過するタイミング違反のあるパス数に基づいて、ノードあたりのレーティングをレポートします。例 7-38 に、report_bottleneck コマンドとオプションを示します。

例 7-38. report_bottleneck コマンド

```
report_bottleneck
[-cmetric <cmetric_name>]
[-details]
[-metric <default|tns|num_paths|num_fpaths|num_fanins|num_fanouts>]
[-panel <panel_name>]
[-stdout]
[<paths>]
```

デフォルトでは、report_bottleneck コマンドは、ワースト 1,000 のセットアップ・パスのレーティングをレポートします。

以下に示すように、デフォルトのメトリックのほかにも、いくつかの「標準」メトリックを選択できます。

- -metric num_fanouts
- -metric tns

また、カスタム・メトリックを作成して、ファンアウト数、ファンイン数、障害パス数、合計パス数、および他のキーパー数の組み合わせに基づいてノードを評価することもできます。解析対象のパスは、get_timing_paths の呼び出しの結果を、report_bottleneck に最後の引数として渡すことによって指定できます。

表 7-35 に、report_bottleneck コマンドのオプションを示します。

表 7-35. report_bottleneck コマンド (1 / 2)	
オプション	説明
-cmetric <cmetric_name>	個々のノードを評価するカスタム・メトリック・ファンクション。
-details	詳細情報 (立ち下がりエッジ数、ファンイン数など) を表示します。
-metric <default tns num_paths num_fpaths num_fanins num_fanouts>	個々のノードの評価に使用するメトリックを指示します。
-panel <panel_name>	パネルに結果を送信し、新しいパネルの名前を指定します。

表 7-35. report_bottleneck コマンド (2 / 2)

オプション	説明
-stdout	レポートを stdout に送信するよう指示します。
<paths>	解析対象のパス。

例 7-39 に、report_bottleneck コマンドを使用してカスタム・メトリックを作成する方法を示します。

例 7-39. report_bottleneck カスタム・メトリック

レポートするパス数を設定

```
set paths [ get_timing_paths -npaths 1000 -setup ]
```

カスタム・メトリックを作成

```
proc report_bottleneck_custom_metric {arg} {
    # 説明：ファンイン数をカスタム・メトリックとして使用
    upvar $arg metric
    set rating $metric(num_fanins)
    return $rating
}
```

カスタム・メトリックの結果をレポート

```
report_bottleneck -cmetric report_bottleneck_custom_metric -panel "Timing
Analysis Bottleneck Report - Custom" $paths
```

report_datasheet

report_datasheet コマンドを使用して、デザイン全体のタイミング特性を要約したデータシート・レポートを生成します。このコマンドは、セットアップ (tsu) 時間、ホールド (th) 時間、clock-to-output (tco) 時間、最小 clock-to-output (mintco) 時間、伝播遅延 (tpd)、および最小伝播遅延 (mintpd) 時間をレポートします。例 7-40 に、report_datasheet コマンドとオプションを示します。

例 7-40. report_datasheet コマンド

```
report_datasheet
[-append]
[-file <name>]
[-stdout]
[panel_name <name>]
```

表 7-36 に、report_datasheet コマンドのオプションを示します。

オプション	説明
-append	現在のレポートを -file オプションで指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むよう指示します。
-stdout	レポートを stdout に送信するよう指示します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

遅延は関係するベース・クロックまたはポートについてレポートされます。1 つのクロックに対して複数のパスが存在する場合、 t_{SU} 、 t_H 、 t_{CO} および t_{PD} には最長パスの最大遅延が利用され、 $mint_{CO}$ および $mint_{PD}$ には最短パスの最小遅延が利用されます。

report_rskm

report_rskm コマンドを使用して、LVDS レシーバのレシーバ・スキュー・マージンを詳細に示すレポートを生成します。

例 7-41 に、report_rskm コマンドとオプションを示します。

例 7-41. report_rskm コマンド

```
report_rskm
[-append]
[-file <name>]
[-panel_name <name>]
[-stdout]
```

表 7-37 に、report_rskm コマンドのオプションを示します。

タイプ名	説明
-append	現在のレポートを -file オプションで指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むよう指示します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。
-stdout	レポートを stdout に送信するよう指示します。

レシーバ入力スキュー・マージン (RSKM) は、LVDS レシーバ・メガファンクションが動作しなくなるまで使用できるタイム・マージンです。RSKM は、時間単位間隔 (TUI) からサンプリング・ウィンドウ (SW) サイズとレシーバ・チャンネル間スキュー (RCCS) を減算した結果として残るトータル・タイム・マージンとして定義されます。これを式で表すと、[計算式 12](#) のようになります。

$$(12) \quad RSKM = \frac{(TUI - SW - RCCS)}{2}$$

時間単位間隔は、LVDS クロック周期 ($1/f_{MAX}$) です。サンプリング・ウィンドウは、データが LVDS レシーバ・メガファンクションによって正常にサンプリングされるように、入力データが安定していなければならない期間です。サンプリング・ウィンドウ・サイズは、デバイスのスピード・グレードによって異なり、RCCS は LVDS レシーバが受けるチャンネル間スキューを反映します。この RCCS には、アップストリーム・トランスミッタのトランスミッタ・チャンネル間スキュー (TCCS) や、トランスミッタとレシーバ間の最大チャンネル間スキューが含まれます。RCCS は、最大入力遅延と最小入力遅延の差に等しくなります。入力遅延が設定されていない場合、RCCS はデフォルトでゼロになります。

report_tccs

report_tccs コマンドを使用して、LVDS トランスミッタのチャンネル間スキュー・マージンを詳細に示すレポートを生成します。

[例 7-42](#) に、report_tccs コマンドとオプションを示します。

例 7-42. report_tccs コマンド

```
report_tccs
[-append]
[-file <name>]
[-panel_name <name>]
[-quiet]
[-stdout]
```

表 7-38 に、report_tccs コマンドのオプションを示します。

タイプ名	説明
-append	現在のレポートを -file オプションで指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むよう指示します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。
-quiet	デザインに LVDS レシーバが存在しない場合は何も出力しないことを指定します。
-stdout	レポートを stdout に送信するよう指示します。

TCCS は、 t_{CO} のばらつきやクロック・スキューなど、最速および最低速出力遷移間のタイミングの差です。

report_path

report_path コマンドを使用して、任意の 2 つのキーパー・ノード間の最長遅延パスを詳細に示すレポートを生成します。

例 7-43 に、report_path コマンドとオプションを示します。

例 7-43. report_path コマンド

```
report_path
[-append]
[-file <name>]
[-from <names>]
[-min_path]
[-npaths <number>]
[-nworst <number>]
[-panel_name <name>]
[-stdout]
[-summary]
[-through <names>]
[-to <names>]
```

表 7-39 に、report_path コマンドのオプションを示します。

タイプ名	説明
-append	現在のレポートを-file オプションで指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル<name> に書き込むよう指示します。
-from <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの始点になります。
-min_path	最小遅延パスを表示します。
-npaths <number>	レポートするパス数を指定します。
-nworst <number>	各エンドポイントについてレポートするパスの最大数を指定します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。
-stdout	レポートを stdout に送信するよう指示します。
-summary	検出された各パスのサマリを含む 1 つのテーブルを作成します。
-through <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。フォルス・パスが<names> を通過することを指定します。
-to <names>	<names> は、デザイン内のオブジェクトのコレクションまたはリストです。<names> は、パスの終点になります。


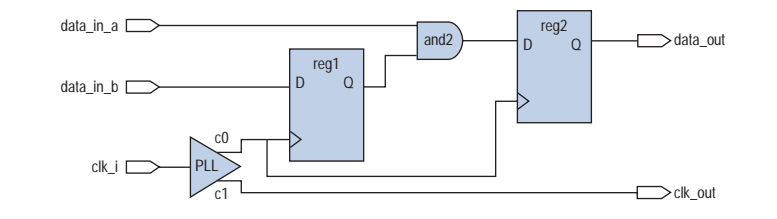
 レポートされる遅延パスは、レジスタやポートなどのキーパー・ノードを通過できません。代わりに、遅延パスはキーパー・ノードの出力ピンからキーパー・ノードの入力ピンまで配置されている必要があります。

図 7-37 に、レジスタ間パスを持つ簡単なデザインを示します。

図 7-37. 簡単なレジスタ間パス



例 7-44 に、次のコマンドから生成されるレポートを示します。

```
report_path -from [get_pins {reg1|regout}] -to [get_pins \
{reg2|datain}] -npaths 1 -panel_name "Report Path" -stdout
```

例 7-44. キーパー出力ピンからキーパー入力ピンまでの report_path

```
Info: =====
Info: From Node :reg1|regout
Info: To Node :reg2|datain
Info:
Info: Path:
Info:
Info: Total (ns) Incr (ns) Type Element
Info: =====
Info: 0.000 0.000 reg1|regout
Info: 0.206 0.206 RR IC and2|datae
Info: 0.360 0.154 RR CELL and2|combout
Info: 0.360 0.000 RR IC reg2|datain
Info:
Info: Total Path Delay : 0.360
Info: =====
```

例 7-45 に、次のコマンドから生成されるレポートを示します。

```
report_path -from [get_ports data_in_a] -to \
[get_pins {reg2|regout}] -npaths 1
```

例 7-45. キーパー間出力ピンからの report_path

```
Info:Report Path:No paths were found
0 0.000
```

例 7-45 にはパスはレポートされていません。これは、デスティネーションがキーパー・ノードの入力ピンを通過するからです。

check_timing

check_timing コマンドを使用して、デザインまたは適用されている制約の潜在的な問題に関するレポートを生成します。すべての check_timing の結果が深刻な問題というわけではありません。結果を調べて、それが望ましいものかどうかを判断する必要があります。例 7-46 に、check_timing コマンドとオプションを示します。

例 7-46. check_timing コマンド

```
check_timing
[-append]
[-file <name>]
[-include <check_list>]
[-stdout]
[-panel_name <name>]
```

表 7-40 に、check_timing コマンドのオプションを示します。

オプション	説明
-append	現在のレポートを-fileオプションで指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル<name>に書き込むよう指示します。
-include	<check_list> を使用してチェックを実行するよう指示します。チェックのリストについては、表 7-41 を参照してください。
-stdout	レポートを stdout に送信するよう指示します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

表 7-41 に、実行可能なチェックを示します。

オプション	説明
no_clock	レジスタのクロック・ピンに少なくとも 1つのクロックがあるかどうかチェックし、クロックとして特定されているポートにクロックが割り当てられているか確認します。
multiple_clock	レジスタのクロック・ピンに多くて 1つのクロックがあるかどうかチェックします。複数のクロックがレジスタのクロック・ピンに到達すると、両方のクロックが解析に使用されます。
generated_clock	生成クロックが有効かどうかチェックします。生成クロックは、有効クロックによって駆動されるソースを持っている必要があります。また、生成クロックはループ内で相互に依存してはなりません (clk2 が clk1 をすでにソースとして使用している場合、clk1 は clk2 をソースにできない)。
no_input_delay	クロックとして特定されていないすべての入力ポートに入力遅延が設定されているかどうかチェックします。
no_output_delay	すべての出力ポートに出力遅延が設定されているかどうかチェックします。

表 7-41. 実行可能なチェック (2 / 2)

オプション	説明
partial_input_delay	入力遅延が完全かどうかチェックします。入力遅延に、rise-min、fall-min、rise-max、および fall-max 部分が設定されていることを確認します。
partial_output_delay	出力遅延が完全かどうかチェックします。出力遅延に、rise-min、fall-min、rise-max、および fall-max 部分が設定されていることを確認します。
reference_pin	reference_pin オプションを使用して set_input_delay および set_output_delay で指定されたりファレンス・ピンが有効かどうかチェックします。reference_pin が有効なのは、同じ set_input_delay/set_output_delay コマンドで指定されるクロック・オプションが、reference_pin のダイレクト・ファンインにあるクロックと一致する場合です。reference_pin のダイレクト・ファンインにあるというのは、クロックと reference_pin の間にキーパーがないという意味です。
latency_override	ポートまたはピンに適用されるクロック・レイテンシ制約は、クロックに設定されているより汎用的なクロック・レイテンシを無効にします。クロックに設定されるクロック・レイテンシは、クロックによって駆動されるすべてのキーパーに適用されます。ピンまたはポートに設定されるクロック・レイテンシは、ポートまたはピンのファン・アウトでレジスタに適用されます。
loops	デザインに、強度に接続されたコンポーネントが存在しないかチェックします。このようなループがあるとデザインが正しく解析されません。ループは存在するがマークされているため横断しないことを指示します。
latches	デザインにラッチがあるかどうかチェックします。Quartus II TimeQuest タイミング・アナライザは、ラッチが存在しており、それらのラッチを適切に解析できないことをユーザーに警告します。

例 7-47 に、check_timing コマンドの使用方を示します。

例 7-47. check_timing コマンド

デザインを制約

```
create_clock -name clk -period 4.000 -waveform { 0.000 2.000 } \
[get_ports clk]
set_input_delay -clock clk2 1.5 [get_ports in*]
set_output_delay -clock clk 1.6 [get_ports out*]
set_false_path -from [get_keepers in] -through [get_nets r1] -to \
[get_keepers out]
```

問題があったかどうかチェック

```
check_timing -include {loops latches no_input_delay partial_input_delay}
```

report_clock_fmax_summary

report_clock_fmax_summary を使用して、ユーザー指定のクロック周期に関係なく、デザインのすべてのクロックについて可能な f_{MAX} をレポートします。 f_{MAX} は、ソースおよびデスティネーション・レジスタまたはポートが同じクロックによってドライブされるパスに関してのみ計算されます。生成クロックなど、異なるクロックのパスは無視されます。クロックとその反転の間のパスについて、 f_{MAX} は、 f_{MAX} は立ち上がりエッジと立ち下がりエッジがあたかも f_{MAX} に従ってスケーリングされ、デューティ・サイクル（パーセンテージで）が維持されるかのように計算されます。

例 7-48 に、report_clock_fmax_summary コマンドとオプションを示します。

例 7-48. report_clock_fmax_summary コマンド

```
report_clock_fmax_summary
[-append]
[-file <name>]
[-panel_name <name>]
[-stdout]
```

表 7-42 に、report_clock_fmax_summary コマンドのオプションを示します。

オプション	説明
-append	現在のレポートを -file オプションで指定されるファイルに追加するよう指定します。
-file <name>	現在のレポートをファイル <name> に書き込むよう指示します。
-stdout	レポートを stdout に送信するよう指示します。
-panel_name <name>	パネルに結果を送信し、新しいパネルの名前を指定します。

f_{MAX} サマリ・レポートには、 f_{MAX} 、Restricted f_{MAX} 、Clock Name、および Note の4つのカラムがあります。表 7-43 に各カラムの説明を示します。

表 7-43. f_{MAX} サマリ・レポート・カラム	
カラム名	説明
f_{MAX}	内部レジスタ間で実行できる、可能な最速の周波数を示します。これがクロック・ポートをドライブできる最速周波数ではありません。
Restricted f_{MAX}	クロック・ポートで実行できる、可能な最速の周波数を示します。ホールド・タイミング要求、クロックの I/O エッジ・レート制限 (I/O 規格にも依存)、レジスタの最小パルス幅チェック、RAM および DSP レジスタの最小周期チェックなど、さまざまな理由からこの数値が、 f_{MAX} カラムよりも低くなる場合があります。
Clock Name	クロック名を示します。
Note	クロックに関連する注意事項を示します。

create_timing_summary

ワースト・ケースのクロック・セットアップおよびクロック・ホールド・スラックと、エンドポイントの TNS (Total Negative Slack) をクロック・ドメインごとにレポートします。Total Negative Slack は、クロック・ドメイン内の各デスティネーション・レジスタまたはポートに対してゼロ未満のすべてのスラックを合計したものです。

例 7-49 に、create_timing_summary コマンドとオプションを示します。

例 7-49. create_timing_summary コマンド

```
create_timing_summary
[-append]
[-file <name>]
[-hold]
[-panel_name <name>]
[-recovery]
[-removal]
[-setup]
[-stdout]
```

表 7-44 に、`create_timing_summary` コマンドのオプションを示します。

オプション	説明
<code>-append</code>	現在のレポートを <code>-file</code> オプションで指定されるファイルに追加するよう指定します。
<code>-file <name></code>	現在のレポートをファイル <code><name></code> に書き込むよう指示します。
<code>-hold</code>	クロック・ホールド・チェック・サマリ・レポートを生成します。
<code>-panel_name <name></code>	パネルに結果を送信し、新しいパネルの名前を指定します。
<code>-recovery</code>	リカバリ・チェック・サマリ・レポートを生成します。
<code>-removal</code>	リムーバル・チェック・サマリ・レポートを生成します。
<code>-setup</code>	クロック・セットアップ・チェック・サマリ・レポートを生成します。
<code>-stdout</code>	レポートを <code>stdout</code> に送信するよう指示します。

タイミング 解析機能

TimeQuest タイミング・アナライザは、デザインのスルー解析を強化および提供する多数の機能をサポートしています。この項では、TimeQuest タイミング・アナライザで使用できる多数の機能について説明します。

マルチ・コーナー解析

マルチ・コーナー解析により、デザインでスタティック・タイミング解析を実行しながら、さまざまな動作条件（電圧、プロセス、および温度）の下でデザインを検証できます。

スタティック・タイミング解析に使用するデバイスの動作条件を変更するには、`set_operating_conditions` コマンドを使用します。



解析するデバイスのスピード・グレードを変更するには、`create_timing_netlist` コマンドを使用します。`create_timing_netlist` コマンドについて詳しくは、7-27 ページの「タイミング・ネットリストの作成」を参照してください。

例 7-50 に、set_operating_conditions コマンドとオプションを示します。

例 7-50. set_operating_conditions コマンド

```
set_operating_conditions
[-model <fast|slow>]
[-speed <speed grade>]
[-temperature <value in °C>]
[-voltage <value in mV>]
[<operating condition Tcl object>]
```

表 7-45 に、set_operating_conditions コマンドのオプションを示します。


表 7-45. set_operating_conditions コマンドのオプション	
オプション	説明
-model <fast slow>	タイミング・モデルを指定します。
-speed <speed grade>	デバイスのスピード・グレードを指定します。
-temperature <value in °C>	動作温度を指定します。
-voltage <value in mV>	動作電圧を指定します。
<operating condition Tcl object>	動作条件を指定する動作条件 Tcl オブジェクトを指定します。



動作条件 Tcl オブジェクトを使用する場合、model、speed、temperature、および voltage オプションは不要です。動作条件 Tcl オブジェクトを使用しない場合、model は必ず指定しますが、-speed、-temperature、および -voltage オプションは省略可能で、該当する場合はデバイスの適切なデフォルト値を使用します。

表 7-46 に、各デバイス・ファミリに設定可能な動作条件をいくつか示します。

デバイス・ファミリ	使用可能な条件				動作条件 Tcl オブジェクト
	スピード・グレード	モデル	電圧 (mV)	温度 (清)	
Stratix III	4	slow slow fast	1100 1100 1100	85 0 0	4_slow_1100mv_85c 4_slow_1100mv_0c MIN_fast_1100mv_0c
Cyclone® III	7	slow slow fast	1200 1200 1200	85 0 0	7_slow_1200mv_85c 7_slow_1200mv_0c MIN_fast_1200mv_0c
Stratix II	4	slow fast	N/A	N/A	4_slow MIN_fast
Cyclone II	6	slow fast	N/A	N/A	6_slow MIN_fast

 ターゲット・デバイスに使用可能な動作条件のリストを取得するには、`get_available_operating_conditions` コマンドを使用します。

例 7-51 に、Stratix III デザインの動作条件として、低速モデル、1100 mV、85°C を設定する方法を示します。

例 7-51. 個々の値での動作条件の設定

```
set_operating_conditions -model slow -temperature 85 -voltage 1100
```

また、例 7-51 に示す動作条件は、例 7-52 に示す Tcl オブジェクトを使用して設定できます。

例 7-52. Tcl オブジェクトによる動作条件の設定

```
set_operating_conditions 4_slow_1100mv_85c
```

アドバンスド I/O タイミングおよびボード・トレース・モデルのアサインメント

Quartus II TimeQuest タイミング・アナライザは、アドバンスド I/O タイミングとボード・トレース・モデルのアサインメントを使用して、デザインの I/O バッファ遅延をモデル化できます。

アドバンスド I/O 機能をオンまたはオフにするには、**Settings** ダイアログ・ボックスの **TimeQuest Timing Analyzer** オプションで、**on** または **off** を選択します。

Advanced I/O Timing をオンまたはオフにするか、ボード・トレース・モデルのアサインメントを変更し、リコンパイルしないでタイミングを解析する場合は、タイミング・ネットリストを作成するときに `-force_dat` コマンドを使用する必要があります。Quartus II TimeQuest タイミング・アナライザの Tcl コンソールで、次のコマンドを入力します。

```
create_timing_netlist -force_dat 1
```

アドバンスド I/O タイミングをオンまたはオフにするか、ボード・トレース・モデルのアサインメントを変更し、リコンパイルしてからタイミングを解析する場合は、タイミング・ネットリストを作成するときに `-force_dat` コマンドを使用する必要はありません。タイミング・ネットリストは、`create_timing_netlist` コマンドを使用するか、**Tasks** ペインの **Create Timing Netlist** タスクを使用して作成できます。



アドバンスド I/O 機能について詳しくは、「[Quartus II ハンドブック Volume 2](#)」の「[I/O 管理](#)」の章を参照してください。


ワイルドカードの割り当てとコレクション

デザインの多数のノードにより簡単に制約を適用するために、Quartus II TimeQuest タイミング・アナライザでは、「*」と「?」をワイルドカード文字として使用できます。これらのワイルドカード文字を使用すると、デザインで指定する必要がある個々の制約数を減らすことができます。

ワイルドカード文字「*」は任意の文字列と一致します。例えば、`reg*` として指定されたノードにアサインメントを行う場合、Quartus II TimeQuest タイミング・アナライザは、先頭の `reg` とその後に続く 0 文字、1 文字、またはいくつかの文字の組み合わせ (例えば、`reg1`、`reg[2]`、`regbank`、`reg12bank`) と一致するデザイン・ノードをすべて検索し、それらにアサインメントを適用します。


ワイルドカード文字「?」は、任意の1文字と一致します。例えば、reg?として指定されたノードにアサインメントを行う場合、Quartus II TimeQuest タイミング・アナライザは、先頭の reg とその後に続く任意の1文字の組み合わせ（例えば、reg1、rega、reg4）と一致するデザイン・ノードをすべて検索し、それらにアサインメントを適用します。

コレクション・コマンドの get_cells と get_pins には、ワイルドカード文字による検索を向上させる3つのオプションがあります。より有効な検索結果を得るには、デフォルト動作、-hierarchical オプション、または -compatibility オプションを選択します。

 パイプ文字を使用すると、Quartus II TimeQuest タイミング・アナライザで、ある階層レベルと次の階層レベルが区切られます。例えば、<absolute full cell name>|<pin suffix> は、階層とピン名を区切る「|」を使用して、階層ピン名を表します。

コレクション・コマンド get_cells と get_pins をオプションなしで使用すると、ピン名の階層レベル単位でデフォルトの検索動作が実行されます。つまり、検索はレベル単位で実行されます。完全階層名には、「|」を使用して階層レベルを区切った複数の階層レベルが含まれていることがあり、各ワイルドカード文字は1つの階層レベルのみを表します。例えば、「*」は最初の階層レベルを表し、「*|*」は最初の階層レベルと2番目の階層レベルを表します。

コレクション・コマンドの get_cells と get_pins を -hierarchical オプション付きで使用すると、<short cell name>|<pin name> 形式の相対的な階層パス名に対して再帰的マッチングが実行されます。検索はノード名に対して実行されます。例えば、階層パスではなく、名前の最後の階層に対して実行されます。デフォルト動作とは異なり、このオプションは検索をパイプ文字で表される各階層レベルに制限しません。

 get_cells -hierarchical オプションによる検索では、パイプ文字は使用できません。ただし、get_pins コレクション検索では、パイプ文字を使用できます。

コレクション・コマンド get_cells と get_pins を -compatibility オプションを指定して使用すると、Quartus II クラシック・タイミング・アナライザと同様の検索が実行されます。このオプションは階層パス全体を検索し、パイプ文字は特殊文字として扱われません。

デザインに以下のセルが存在すると仮定します。

```
foo
foo|bar
```

また、次のピン名を仮定とします。

```
foo|dataa
foo|datab
foo|bar|datac
foo|bar|datad
```

表 7-47 に、これらの検索文字列を使用した結果を示します。

検索文字列	検索結果
get_pins * dataa	foo dataa
get_pins * datac	<empty>
get_pins * * datac	foo bar datac
get_pins foo* *	foo dataa, foo datab
get_pins -hierarchical * * datac	<empty> (1)
get_pins -hierarchical foo *	foo dataa, foo datab
get_pins -hierarchical * datac	foo bar datac
get_pins -hierarchical foo * datac	<empty> (1)
get_pins -compatibility * datac	foo bar datac
get_pins -compatibility * * datac	foo bar datac

表 7-47 の注:

(1) 検索文字列に *|*| を加えたため、検索結果は <empty> になります。

デザインのリセット

解析中のデザインからタイミング制約と例外をすべて削除するには、`reset_design` コマンドを使用します。このコマンドは、クロック、生成クロック、派生クロック、入力遅延、出力遅延、クロック・レイテンシ、クロック不確実性、クロック・グループ、フォルス・パス、マルチサイクル・パス、最小遅延、および最大遅延をすべて削除します。

このコマンドを使用すると、タイミング・ネットリストを削除して新たに再作成しなくても、容易に解析の初期状態に戻すことができます。

クロス・プロービング

クロス・プロービング機能により、TimeQuest タイミング・アナライザから Quartus II ソフトウェアで使用可能な各種ツール（または、その逆）へのパスとエレメントを探索できます。

TimeQuest GUI で、**View** ペインの任意のパスを右クリックして、**Locate Path** または **Locate** を選択できます。

ソースは **From Node** カラムのエレメントで、デスティネーションは **To Node** カラムのエレメントです。

Locate Path オプションにより、現在選択しているロウのデータ到達パス（デフォルト）を探索できます。データ要求時間パスを探索するには、データ要求パス・パネルでロウを選択します。



Locate Required Path コマンドは、表示するパスがある場合にのみ使用できます。ユーザーがクロック・パスもレポートしない限り、要求パスにはおそらく1つのノードしか存在しません。この場合、コマンドは使用できません。

Locate オプションを使用すると、ハイライトしたエレメントを探索できます。

Locate Path および **Locate** コマンドは、Chip Planner、Technology Map Viewer、または Resource Property Editor へのクロス・プローブが可能です。また、**Locate Path** オプションは、Critical Path Settings へのクロス・プローブも可能です。

Chip Planner の **Critical Path Settings** ダイアログ・ボックスから、TimeQuest タイミング・アナライザへのクロス・プローブを行って、デザイン内のクリティカル・パスをレポートできます。

locate

Console ペインで `locate` コマンドを使用して、Chip Editor、Critical Path Settings、Resource Property Editor、および Technology Map Viewer へのクロス・プローブを行います。

例 7-53 に、`locate` コマンドとオプションを示します。

例 7-53. locate コマンド

```
locate
[-chip]
[-color <black|blue|brown|green|grey|light_grey|orange|purple|red|white>]
[-cps]
[-label <label>]
[-rpe]
[-tmv]
<items>
```

表 7-48 に、locate コマンドのオプションを示します。

オプション	説明
-chip	Chip Planner でオブジェクトを探索します。
-color <black blue brown green grey light_grey orange purple red white>	探索するオブジェクトを識別します。
-cps	Chip Planner の Critical Path Settings ダイアログでオブジェクトを探索します。
-label <label>	探索するオブジェクトの識別用ラベルを指定します。
-rpe	Resource Property Editor で探索します。
-tmv	Technology Map Viewer でオブジェクトを探索します。
<items>	探索するアイテム。コレクションまたはオブジェクト（パス、ポイント、ノード、ネット、キーパー、レジスタなど）は、対応するコレクションまたはオブジェクトへの参照を渡すことによって探索できます。

例 7-54 に、TimeQuest タイミング・アナライザから Chip Editor に 10 個のパスをクロス・プローブし、Technology Map Viewer ですべてのデータ・ポートを探索する方法を示します。

例 7-54. TimeQuest からのクロス・プロービング

```
# Chip Editor への 10 個の最長パスにあるすべてのノードを  
# 探索
```

```
locate [get_path -npaths 10] -chip
```

```
# Tech Map Viewer へのデータで開始するすべてのポートを探索
```

```
locate [get_ports data*] -tmv
```

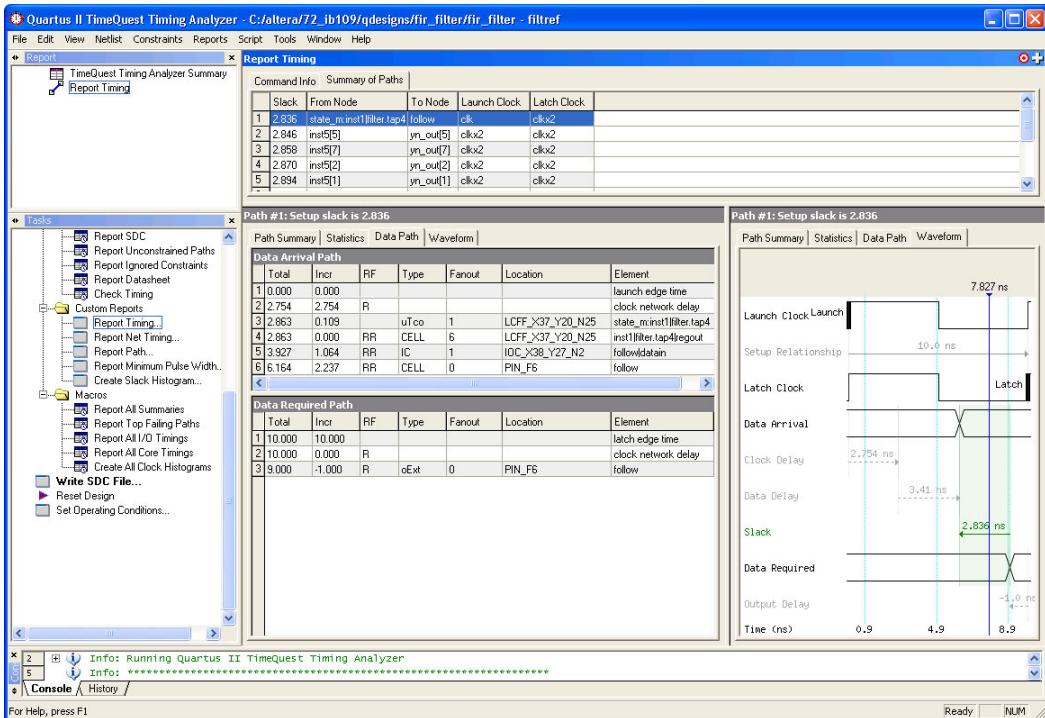
TimeQuest タイミング・ アナライザ GUI

Quartus II TimeQuest タイミング・アナライザでは、直観的で使いやすい GUI からデザインを効率的に制約および解析できます。この GUI は以下のペインで構成されます。

- 「Quartus II ソフトウェアのインタフェースおよびオプション」 7-101 ページを参照。
- 「View ペイン」 7-101 ページを参照。
- 「Tasks ペイン」 7-104 ページを参照。
- 「Console ペイン」 7-107 ページを参照。
- 「Report ペイン」 7-107 ページを参照。
- 「Constraints」 7-108 ページを参照。
- 「Name Finder」 7-110 ページを参照。
- 「Target ペイン」 7-111 ページを参照。
- 「SDC エディタ」 7-112 ページを参照。

各ペインは生産性を高める機能を提供します (図 7-38)。

図 7-38. TimeQuest GUI




Quartus II ソフトウェアのインタフェースおよびオプション

Quartus II ソフトウェアでは、デザインの Compilation Report に生成される Quartus II TimeQuest タイミング・アナライザ・レポートのために各種オプションをコンフィギュレーションできます。

Settings ダイアログ・ボックスの TimeQuest タイミング・アナライザ設定では、表 7-49 に示すオプションをコンフィギュレーションできます。

オプション	説明
SDC files to include in the project	プロジェクトに関連付けられている SDC ファイルを追加および削除します。
Enable Advanced I/O Timing	各ピンに指定されているボード・トレース・モデルから、アドバンスド I/O タイミング結果を生成します。
Enable multicornertiming analysis during compilation	ターゲット・デバイスの使用可能なすべての動作条件について複数のレポートを生成します。
Report worst-case paths during compilation	クロック・ドメインあたりのワースト・ケース・パス・レポートを生成します。
Tcl Script File for customizing report during compilation	カスタム・レポート生成のソースとなる任意のカスタム・スクリプトを指定します。

 表 7-49 に示すオプションは、Compilation Report で生成するレポートのみをコントロールするものであり、Quartus II TimeQuest タイミング・アナライザで生成するレポートはコントロールしません。

View ペイン

View ペインは、タイミング解析結果のメイン表示領域です。**View** ペインを使用して、サマリ・レポート、カスタム・レポート、またはヒストグラムを表示します。図 7-39 に、**Report** ペインから **Summary (Setup)** レポートを選択した後の **View** ペインを示します。

図 7-39. Summary (Setup) レポート

Summary (Setup)			
	Clock	Slack	End Point TNS
1	inclk_0	0.303	0.000
2	altpll0:instaltpll:altpll_component_clk_0	0.384	0.000

View ペイン : 分割

タイミング結果を適切に解析するには、複数のレポートを比較することがきわめて重要です。複数のレポートを表示しやすいように、**View** ペインではウィンドウを分割することができます。ウィンドウを分割すると、**View** ペインは複数のウィンドウに分かれるため、いくつかのレポートを横に並べて表示できます。

View ペインを複数のウィンドウに分割するには、**View** ペインの右上隅にある分割アイコンを使用します。このアイコンを別の方向にドラッグすると、**View** ペインにウィンドウ・ビューが生成されます。例えば、分割アイコンを左にドラッグすると、現在のウィンドウの右側に新しいウィンドウが作成されます (図 7-40)。

図 7-40. View ペインを左に分割 (左に分割する前および後)

Summary (Setup)			
	Clock	Slack	End Point TNS
1	inclk_0	0.303	0.000
2	altpll0:instaltpll:altpll_component_clk_0	0.384	0.000

Summary (Setup)				Summary (Setup)			
	Clock	Slack	End Point TNS		Clock	Slack	End Point TNS
1	inclk_0	0.303	0.000	1	inclk_0	0.303	0.000
2	altpll0:instaltpll:altpll_component_clk_0	0.384	0.000	2	altpll0:instaltpll:altpll_component_clk_0	0.384	0.000

分割アイコンを斜めにドラッグすると、**View** ペインに新しい3つのウィンドウが作成されます (図 7-41)。

図 7-41. View ペインを斜めに分割 (斜めに分割する前および後)

Summary (Setup)			
	Clock	Slack	End Point TNS
1	inclk0	0.303	0.000
2	altpll0:inst altpll:altpll_component_clk0	0.384	0.000

Summary (Setup)			
	Clock	Slack	End Point TNS
1	inclk0	0.303	0.000
2	altpll0:inst altpll:altpll_component_clk0	0.384	0.000

Summary (Setup)			
	Clock	Slack	End Point TNS
1	inclk0	0.303	0.000
2	altpll0:inst altpll:altpll_component_clk0	0.384	0.000

Summary (Setup)			
	Clock	Slack	End Point TNS
1	inclk0	0.303	0.000
2	altpll0:inst altpll:altpll_component_clk0	0.384	0.000

分割アイコンを下にドラッグすると、現在のウィンドウのすぐ下に新しいウィンドウが作成されます。

View ペイン : 分割したウィンドウの削除

分割アイコンを使用して **View** ペインに作成したウィンドウを削除するには、ウィンドウの枠線を、削除するウィンドウの上までドラッグします (図 7-42)。

図 7-42. 分割したウィンドウの削除（分割を削除する前および後）

The image shows two states of the Quartus II interface. The top state shows two windows: 'Summary (Setup)' and 'Fmax Summary'. The bottom state shows only the 'Summary (Setup)' window, indicating that the 'Fmax Summary' window has been removed.

	Clock	Slack	End Point TNS
1	inclk0	0.303	0.000
2	altpll0:inst altpll:altpll_component_clk0	0.384	0.000

	Fmax (MHz)	Clock Name
1	1623.38	altpll0:inst altpll:altpll_component_clk0
2	1434.72	inclk0

This panel reports FMAX for every clock in the design, regardless of the user-specified clock periods. FMAX is only computed for paths where the source and destination registers or ports are driven by the same clock. Paths of different clocks, including generated clocks, are ignored. For paths


	Clock	Slack	End Point TNS
1	inclk0	0.303	0.000
2	altpll0:inst altpll:altpll_component_clk0	0.384	0.000

Tasks ペイン

Tasks ペインを使用して、ネットリスト・セットアップやレポート生成などの一般的なコマンドにアクセスします。

Tasks ペインには、**Open Project**、**Set Operating Conditions**、および **Reset Design** という一般的なコマンドがあります。ネットリスト・セットアップやレポート生成などの他のコマンドは、以下のフォルダにあります。

- Netlist Setup
- Reports

 **Tasks** ペインの各コマンドには同等の Tcl コマンドがあり、コマンドの実行時に **Console** ペインに表示されます。

プロジェクトのオープンと Synopsys Design Constraints ファイルの記述


Quartus II TimeQuest タイミング・アナライザでプロジェクトを開くには、**Open Project** タスクをダブルクリックします。Quartus II TimeQuest タイミング・アナライザを Quartus II ソフトウェア GUI から起動すると、プロジェクトは自動的に開きます。

Quartus II TimeQuest タイミング・アナライザで初期 SDC ファイルを読み出した後、タイミング・ネットリストに制約を追加、またはそれから削除できます。初期 SDC ファイルは読み出されると、Quartus II TimeQuest タイミング・アナライザの制約よりも古くなります。**Write SDC File** コマンドを使用して、Quartus II TimeQuest タイミング・アナライザの制約の現在の状態を反映した最新の SDC ファイルを生成します。

Netlist Setup フォルダ

Netlist Setup フォルダには、タイミング解析のためにタイミング・ネットリストを設定するのに使用されるタスクが含まれます。このフォルダには、**Create Timing Netlist**、**Read SDC File**、および **Update Timing Netlist** の3つのタスクがあります。

Create Timing Netlist タスクを使用して、Quartus II TimeQuest タイミング・アナライザでスタティック・タイミング解析を実行する際に使用するネットリストを作成します。このネットリストは、Quartus II TimeQuest タイミング・アナライザで、タイミング解析のためにのみ使用されます。

 タイミング・ネットリストは、Quartus II TimeQuest タイミング・アナライザで解析を実行する前に必ず作成する必要があります。

Read SDC File コマンドを使用して、タイミング・ネットリストに制約を適用します。デフォルトでは、**Read SDC File** コマンドは `<current revision>.sdc` ファイルを読み出します。

 `read_sdc` コマンドを使用して、デザインの現在のリビジョンに関連付けられていない SDC ファイルを読み出します。

Update Timing Netlist コマンドを使用して、制約の入力後または SDC ファイルの読み出し後に、タイミング・ネットリストを更新します。デザインに制約を追加したり、それから削除する場合は、このコマンドを使用する必要があります。

Reports フォルダ

Reports フォルダには、スタティック・タイミング解析結果のタイミング・サマリ・レポートを生成するためのコマンドが含まれています。このフォルダ内にある 12 のコマンドを表 7-50 にまとめます。

レポート・タスク	説明
Report f_{MAX} Summary	デザイン内のすべてのクロックの f_{MAX} サマリ・レポートを生成します。
Report Setup Summary	デザイン内のすべてのクロックについてのクロック・セットアップ・サマリ・レポートを生成します。
Report Hold Summary	デザイン内のすべてのクロックについてのクロック・ホールド・サマリ・レポートを生成します。
Report Recovery Summary	デザイン内のすべてのクロックについてのリカバリ・サマリ・レポートを生成します。
Report Removal Summary	デザイン内のすべてのクロックについてのリムーバル・サマリ・レポートを生成します。
Report Clocks	デザイン内で作成されるすべてのクロックについてのサマリ・レポートを生成します。
Report Clock Transfers	デザイン内で検出されるすべてのクロック転送についてのサマリ・レポートを生成します。
Report Minimum Pulse Width	デザイン内のすべての最小パルス幅についてのサマリ・レポートを生成します。
Report SDC	SDC ファイルから読み出す制約についてのサマリ・レポートを生成します。
Report Unconstrained Paths	デザイン内の制約されていないすべてのパスについてのサマリ・レポートを生成します。
Report Ignored Constraints	デザイン用の SDC 制約のうち、無視されたすべての SDC 制約についてのサマリ・レポートを生成します。
Report Datasheet	デザインのデータシート・レポートを生成します。

Macros フォルダ

Macros フォルダには、Quartus II TimeQuest タイミング・アナライザ・ユーティリティ・パッケージで使用可能なカスタム・タスクを実行するためのコマンドが含まれています。このようなコマンドとして、**Report All Summaries**、**Report Top Failing Paths**、および **Create All Clock Histograms** があります。

表 7-51 に、Macros フォルダで使用可能なコマンドを示します。

マクロ・タスク	説明
Report All Summaries	このコマンドは、 Report Setup Summary 、 Report Hold Summary 、 Report Recovery Summary 、 Report Removal Summary 、および Minimum Pulse Width コマンドを実行して、すべてのサマリ・レポートを生成します。
Report Top Failing Paths	このコマンドは、タイミング違反のあるトップ・パスのリストを含むレポートを生成します。
Create All Clock Histograms	このコマンドは、 Create Slack Histogram コマンドを実行して、デザイン内のすべてのクロックのクロック・ヒストグラムを生成します。
Report All I/O Timings	このコマンドは、デバイス・ポートで開始または終了するすべてのタイミング・パスのレポートを生成します。
Report All Core Timings	このコマンドは、デバイス・レジスタで開始および終了するすべてのタイミング・パスのレポートを生成します。

Console ペイン

Console ペインは、Quartus II TimeQuest タイミング・アナライザのメッセージ・センターとしての機能と、対話型 Tcl コンソールとして機能を持ちます。**Console** ペインには、**Console** および **History** という 2 つのタブがあります。**Console** タブには、情報メッセージや警告メッセージなど、すべてのメッセージが表示されます。また、**Console** タブでは、Synopsys デザイン制約と Tcl コマンドを入力して実行することもできます。さらに、**Console** タブには、**Tasks** ペインで実行するすべてのコマンドと同等の Tcl コマンドが表示されます。**History** タブでは、実行される Synopsys デザイン制約と Tcl コマンドをすべて記録します。



タイミング・ネットリストの更新後に **History** タブのコマンドを実行するには、実行するコマンドを右クリックし、**Rerun** をクリックします。

Console および **History** タブの Tcl コマンドをコピーすると、タイミング解析を実行する Tcl スクリプトを簡単に生成できます。

Report ペイン

Report ペインを使用して、**Tasks** ペインおよびカスタム・レポート・コマンドによって生成されたすべてのレポートにアクセスします。**Report** ペインのレポートを選択すると、そのレポートは、**View** ペインのアクティブ・ウィンドウに表示されます。



レポートが現在の制約と比較して古くなっている場合は、レポートの横に「?」アイコンが表示されます。

Constraints

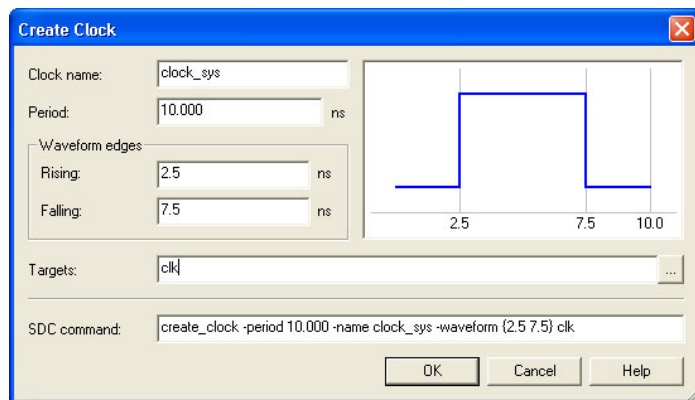
Constraints メニューを使用して、一般的に使用する制約、例外、およびコマンドにアクセスします。このメニューには、ツールバーの **Edit** をクリックし、**Constraint menu** をクリックしてアクセスできます。

Constraints メニューでは、以下のコマンドを使用できます。

- **Create Clock**
- **Create Generated Clock**
- **Set Clock Latency**
- **Set Clock Uncertainty**
- **Set Clock Groups**
- **Remove Clock**

例えば、**Create Clock** ダイアログ・ボックスを使用して、デザインのクロックを作成します。図 7-43 に、**Create Clock** ダイアログ・ボックスを示します。

図 7-43. Create Clock ダイアログ・ボックス



以下のコマンドは、タイミング例外を指定します。これらのコマンドは、Constraints メニューで使用できます。

- **Set False Path**
- **Set Multicycle Path**
- **Set Maximum Delay**

■ Set Minimum Delay

これらのコマンドからタイミング制約または例外を指定する際に使用するダイアログ・ボックスには、必ず SDC コマンド・フィールドがあります。このフィールドには、OK をクリックすると実行される SDC コマンドが含まれます。



Quartus II TimeQuest タイミング・アナライザ・ユーザー・インタフェースで作成されるコマンドと制約は、すべて **Console** ペインにエコーされます。

Constraints メニュー・コマンドで指定する制約は、現在の SDC ファイルに自動的に保存されません。制約を保存するには、**Write SDC File** コマンドを実行する必要があります。

Quartus II TimeQuest タイミング・アナライザの Constraints メニューでは、以下のコマンドを使用できます。

- **Generate SDC File from QSF**
- **Read SDC File**
- **Write SDC File**

Generate SDC File from QSF コマンドは、QSF ファイル内の Quartus II クラシック・タイミング・アナライザ制約を、Quartus II TimeQuest タイミング・アナライザ用の SDC ファイルに変換する Tcl スクリプトを実行します。<current revision>.sdc は、このコマンドによって作成されます。



Generate SDC File from QSF コマンドについては、「Quartus II ハンドブック Volume 3」の「[Switching to the Quartus II TimeQuest Timing Analyzer](#)」の章を参照してください。

Generate SDC File from QSF コマンドは、QSF ファイル内のすべてのタイミング制約および例外を、同等の SDC ファイル制約に変換しようとしています。ただし、すべての QSF ファイルが SDC ファイル制約に変換できるわけではありません。SDC ファイルの作成後はファイルを調べて、すべての制約が正常に変換されたことを確認します。

Read SDC File コマンドは、<current revision>.sdc ファイルを読み出します。

Write SDC File コマンドを選択すると、Quartus II TimeQuest タイミング・アナライザの制約の現在の状態を反映した最新の SDC ファイルが生成されます。

Name Finder

Name Finder ダイアログ・ボックスを使用して、Quartus II TimeQuest タイミング・アナライザ GUI で、制約または例外のターゲットを選択します。Name Finder では、コレクション、フィルタ、およびフィルタ・オプションを指定できます。**Name Finder** ダイアログ・ボックスの **Collections** フィールドには、選択する名前の種類を指定できます。種類を選択するには、**Collection** リストで、以下のリストから希望のコレクション API を選択します。

- `get_cells`
- `get_clocks`
- `get_keepers`
- `get_nets`
- `get_nodes`
- `get_pins`
- `get_ports`
- `get_registers`

各種コレクション API については、7-29 ページの「コレクション」を参照してください。

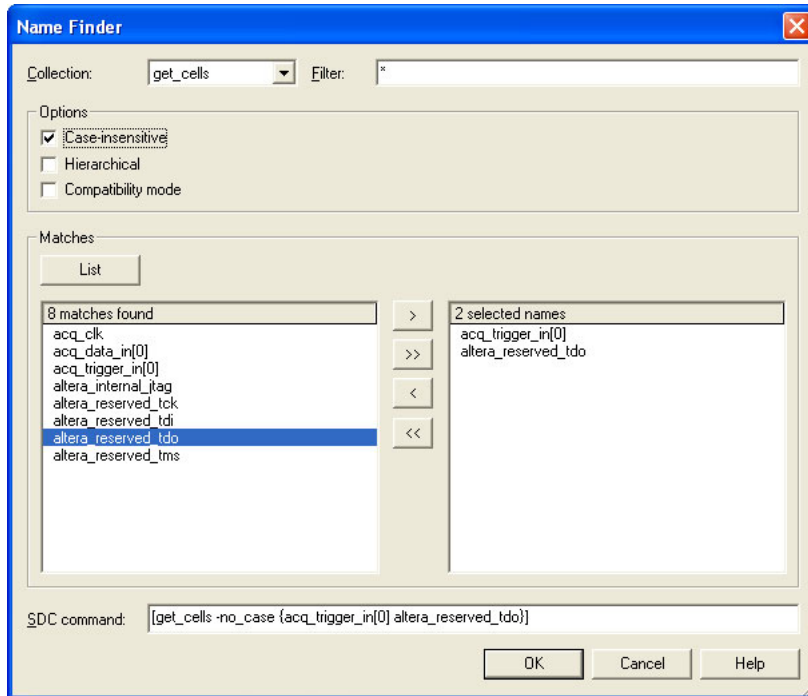
Filter フィールドを使用すると、ワイルドカード文字の使用も含めた独自の基準に基づいて名前をフィルタできます。以下のフィルタ・オプションを使用すると、結果をさらに絞り込むことができます。

- **Case-insensitive**
- **Hierarchical**
- **Compatibility mode**

フィルタ・オプションについては、7-95 ページの「ワイルドカードの割り当てとコレクション」を参照してください。

また、**Name Finder** ダイアログ・ボックスには **SDC command** というフィールドもあり、現在選択されている名前検索コマンドが表示されます。このフィールドの値をコピーして、別の制約ターゲットのフィールドで使用することができます。**Name Finder** ダイアログ・ボックスを図 7-44 に示します。

図 7-44. Name Finder ダイアログ・ボックス



Target ペイン

TimeQuest GUI を使用する場合は、**View** ペインを複数のウィンドウに分割できます。この分割機能を使用すると、**View** ペインに複数のレポートを表示できます。分割後の **View** ペインでは、最後にアクティブになっていたウィンドウに新しいレポートが表示されます。この動作を変更するには、分割されている各ウィンドウの状態を変更します。ウィンドウの状態を変更するには、右上隅にあるターゲット・マークをクリックします (図 7-45)。表 7-52 に、各ウィンドウの状態を示します。

図 7-45. Target ペイン

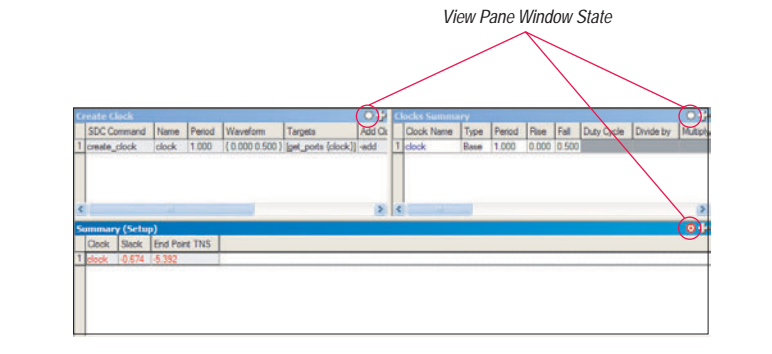


表 7-52. View ペインのウィンドウの状態

状態	説明
Partially Filled Red Circle	アクティブ・ウィンドウに新しいレポートが表示されていることを示します。
Fully Filled Red Circle	アクティブ・ウィンドウであるかどうかに関係なく、そのウィンドウに新しいレポートが表示されていることを示します。
Empty Circle	そのウィンドウに新しいレポートが表示されていないことを示します。

アクティブ・ウィンドウの右上隅にあるターゲット・マークをクリックすると、ウィンドウの状態が変化します。

SDC エディタ

TimeQuest GUI には SDC エディタもあります。SDC エディタを使用すると、簡単かつ便利な方法で SDC ファイルを記述、編集、および表示できます。SDC エディタはコンテキスト依存です。SDC 制約または例外を入力すると、ツールチップが表示され制約または例外に関するオプションやフォーマットを示します。



メニュー・バーの Constraints メニューから **Constraints** ダイアログ・ボックスを開きます。必要なすべてのパラメータを入力すると、SDC は現在のカーソル位置に挿入されます。

まとめ

Quartus II TimeQuest タイミング・アナライザは、複雑なデザイン要件に対応しており、直観的なユーザー・インタフェース、業界標準の制約フォーマットのサポート、およびスクリプト機能を通じて生産性と効率を向上させます。Quartus II TimeQuest タイミング・アナライザは、業界標準の SDC フォーマットをサポートする次世代のタイミング解析ツールで、設計者は複雑なタイミング制約を作成、管理、および解析して、高度なタイミング検証を実行できます。

参考資料

この章では以下のドキュメントを参照しています。

- 「Quartus II ハンドブック Volume 2」の「I/O Management」の章
- 「SDC and TimeQuest API Reference Manual」
- 「Quartus II ハンドブック Volume 4」の「Volume 4: SOPC Builder」
- 「Quartus II ハンドブック Volume 3」の「Quartus II TimeQuest Timing Analyzer」の章
- 「TimeQuest Quick Start Tutorial」

改訂履歴

表 7-53 に、本資料の改訂履歴を示します。

表 7-53. 改訂履歴 (1 / 2)		
日付およびバージョン	変更内容	概要
2008 年 5 月 v8.0.0	<p>Quartus II ソフトウェア v8.0 のための更新。以下のとおり。</p> <ul style="list-style-type: none"> ● 見出し「タイミング要求を指定する」を7-27ページの「Quartus II TimeQuest タイミング・アナライザのフローに関するガイドライン」に変更。 ● 7-31 ページの「SDC 制約ファイル」に順序が重要であることを示す情報を追加。 ● 7-20 ページの「メタスタビリティ」に新しい項を追加。 ● 7-23 ページの「コモン・クロック・パス・ベシミズム」に新しい項を追加。 ● 7-45 ページの「クロック・グループ」から非同期クロックに関する情報を削除。 ● 例 7-28 の情報を更新。 ● 表 7-22 に 3 つの項目を追加。 ● 表 7-24 に情報を追加 (ADoQS 10001445)。 ● 7-83 ページの「report_rskm」に RSKM に関する情報 (数式 (計算式 12) など) を追加。 ● 7-45 ページの「クロック・グループ」の項を追加。 ● 7-90 ページの「report_clock_fmax_summary」に表 7-44 を追加。 ● 表 7-46 の概要に修飾子を追加。 ● 表 7-46 にスピード・グレードの情報を追加。 ● <code>derive_clock_uncertainty</code> コマンドに関する情報 (7-49 ページの「クロック不確実性の自動計算」) から <code>[-dtw]</code> を削除し、<code>[-add]</code> を追加。 ● 7-70 ページの「report_metastability」の項を追加。 ● 7-83 ページの「report_rskm」に RSKM に関する新しい情報を追加。 ● 7-98 ページの「クロス・プロービング」の項を追加。 ● 編集上の些細な変更。 ● 章全体を通して、参照資料にハイパーリンクを追加。 	Quartus II ソフトウェア v8.0 のリリースによる大幅な更新。
2007 年 10 月 v7.2.0	<p>Quartus II ソフトウェア v7.2 のための更新。以下のとおり。</p> <ul style="list-style-type: none"> ● 「Compilation Flow with TimeQuest Guidelines」, 「Timing Analysis Overview」, および 「Specify Design Timing Requirements」の項の編成フローを更新。 ● Clock as Data 解析に関する新しい情報を追加。 	Quartus II ソフトウェア v7.2 のための更新。

日付およびバージョン	変更内容	概要
2007年5月 v7.1.0	<p>Quartus II ソフトウェア v7.1 のための更新。以下のとおり。</p> <ul style="list-style-type: none"> ● ページ 6-57 の「Timing Reports」に report_path のサポートを追加。 ● report_timing 情報を追加 (特にページ 6-11)。 ● 以下の見出しの下に新しい情報を追加。 <ul style="list-style-type: none"> ● 「Derive Clock Uncertainty」 (ページ 6-40) ● 「report_rskm」 (ページ 6-69) ● 「report_tccs」 (ページ 6-69) ● 「report_path」 (ページ 6-70) ● 「Fast Timing Model Analysis」の項を「Multi-Corner Analysis」に置き換え (ページ 6-76)。 ● 7.1 に関する一般的な更新。 	Quartus II ソフトウェア v7.2 のための更新。
2007年3月 v7.0.0	Quartus II ソフトウェア 7.0 のリビジョンおよび日付のみ更新。その他の変更はありません。	—
2006年11月 v6.1.0	<p>Quartus II ソフトウェア v6.1 のための更新。以下のとおり。</p> <ul style="list-style-type: none"> ● 「Specifying Clock Requirements」、「Specifying Input and Output Port Requirements」、および「Reporting」の項に新しい項「Getting Started」を追加 (Create Clock および Create Generated Clock ダイアログ・ボックス / コマンドの説明を含む)。 ● SDC エディタ ● GUI の使いやすさが向上。 ● SDC サポートの更新。 ● 章全体を通していくつかの変更。 	Quartus II ソフトウェア v6.1 のための更新。
2006年7月 v6.0.1	<p>Quartus II ソフトウェア v6.0.1 のための更新。</p> <ul style="list-style-type: none"> ● report_clock_transfers コマンドのタイポ (ページ 6-15) 	—
2006年5月 v6.0.0	初版	—

