

# Vivado Design Suite ユーザー ガイド

## インプリメンテーション

UG904 (v2012.2) 2012 年 8 月 20 日



#### Notice of Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at <http://www.xilinx.com/warranty.htm>; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: <http://www.xilinx.com/warranty.htm#critapps>.

© Copyright 2012 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.

本資料は英語版 (v2012.2) を翻訳したもので、内容に相違が生じる場合には原文を優先します。

資料によっては英語版の更新に対応していないものがあります。

日本語版は参考用としてご使用の上、最新情報につきましては、必ず最新英語版をご参照ください。

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

## 改訂履歴

次の表に、この文書の改訂履歴を示します。

日付	バージョン	改訂内容
2012 年 7 月 25 日	2012.2	初版
2012 年 8 月 20 日	2012.2	route_design コマンドの説明に再配線およびタイミング制約に関する詳細を追加

# 目次

## 第 1 章：インプリメンテーション プロセス

インプリメンテーションの概要 .....	5
インプリメンテーションの前に .....	7
プロジェクト モードと非プロジェクトモード .....	7
RTL および合成済みデザイン .....	8
制約 .....	9
デザイン チェックポイント .....	10
非プロジェクト モードでのインプリメンテーションの実行 .....	11
非プロジェクト モードのサンプル スクリプト .....	11
サンプル スクリプトの詳細 .....	12
プロジェクト モードでのインプリメンテーションの実行 .....	13
インプリメンテーション run の作成 .....	13
インプリメンテーション ストラテジのカスタマイズ .....	20
run の実行 .....	24
インプリメンテーションの段階ごとの実行 .....	25
インプリメンテーション run の監視 .....	26
プロジェクト ステータスの確認 .....	27
インプリメンテーション完了後の次の操作 .....	28
メッセージの表示 .....	29
インプリメンテーション レポートの表示 .....	30

## 第 2 章：インプリメンテーション コマンド

概要 .....	34
合成済みデザインを開く .....	35
synth_design .....	35
read_checkpoint .....	36
open_run .....	36
link_design .....	36
ロジック最適化 .....	37
opt_design .....	37
ロジック最適化制約 .....	38
消費電力の最適化 .....	38
power_opt_design .....	39
配置 .....	39
place_design .....	40
物理合成 .....	41
phys_opt_design .....	41
配線 .....	42
route_design .....	42

## 付録 A：リモート ホストの使用

リモート Linux ホストでの run の起動 .....	44
リモート ホストの設定 .....	44
SSH の設定 .....	47

## 付録 B: その他のリソース

ザイリンクス リソース .....	48
ソリューション センター .....	48
参考資料 .....	48

# インプリメンテーション プロセス

## インプリメンテーションの概要

Vivado™ Design Suite では、図 1-1 に示すように、RTL デザイン、ネットリスト デザイン、IP 中心のデザイン フローを使用して、さまざまなソースからのザイリンクス 7 シリーズ FPGA デザインをインプリメントできます。Vivado ツールでサポートされる異なるデザイン フローを理解するには、『Vivado Design Suite ユーザー ガイド：デザイン フローの概要』(UG892) を参照してください。Vivado インプリメンテーションは、デザインの論理制約、物理制約、タイミング制約を満たしながらネットリストを FPGA デバイスに配置配線するためのすべての手順を含みます。

Vivado インプリメンテーションは、デザイン要件および制限の指定に業界標準の SDC (Synopsys Design Constraint) コマンドをサポートするタイミングドリブン フローです。ザイリンクス デザイン制約 (XDC) 形式のコマンドも追加されています。

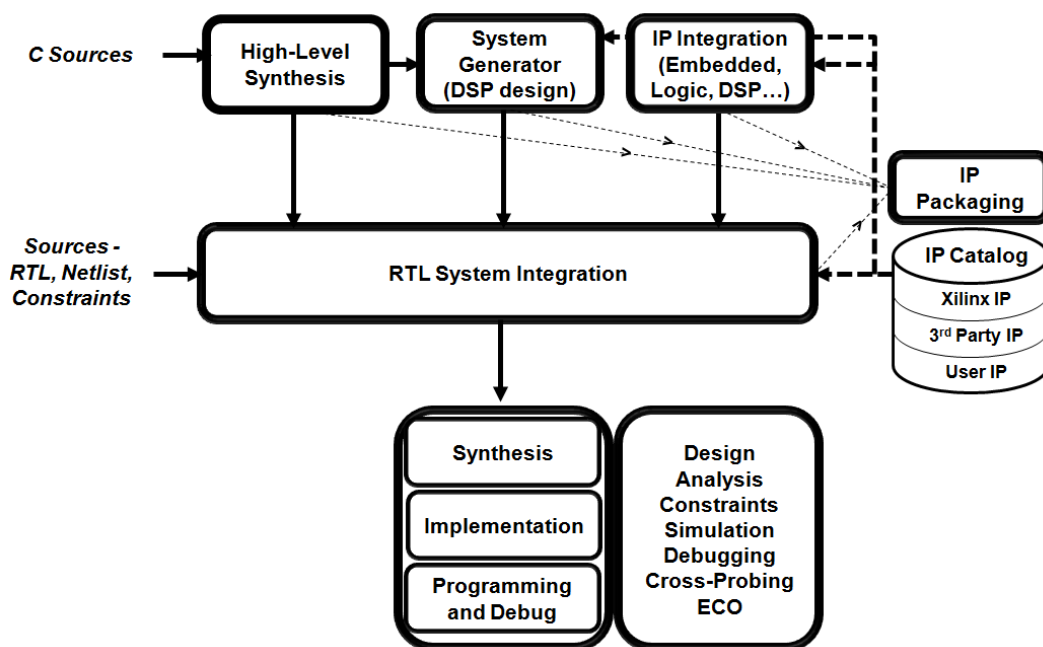


図 1-1 : Vivado Design Suite の高レベル デザイン フロー

Vivado インプリメンテーションは、デザインの論理的な変換および物理的な変換を含み、次のサブ プロセスで構成されています。

1. デザインの最適化：ザイリンクス FPGA にフィットしやすいように論理デザインを最適化します。
2. デザインの消費電力最適化：インプリメント済み FPGA の消費電力を削減するため、デザイン エレメントを最適化します (オプション)。
3. デザインの配置：デザインをターゲット ザイリンクス デバイスに配置します。
4. デザインの物理最適化：ファンアウトの大きいネットのドライバーを複製してロードを分散することにより、デザインのタイミングを最適化します (オプション)。
5. デザインの配線：デザインをターゲット ザイリンクス デバイスに配線します。
6. ビットストリームの生成：ザイリンクス デバイス コンフィギュレーションのビットストリームを生成します。

デザイン フロー全体が Vivado Integrated Design Environment (IDE) に組み込まれており、Flow Navigator というインターフェイスを使用して FPGA デザインおよび IP を作成、インプリメント、および検証できます。デザイン フローを簡略化するため、Flow Navigator でインプリメンテーション プロセス全体をコマンドをクリックするだけで実行できます (図 1-2)。このガイドでは、インプリメンテーション以外の Vivado IDE の詳細は説明しません。FPGA デザイン フロー全体に関して Vivado IDE を理解するには、『Vivado Design Suite ユーザー ガイド : Vivado IDE の使用』(UG893) を参照してください。

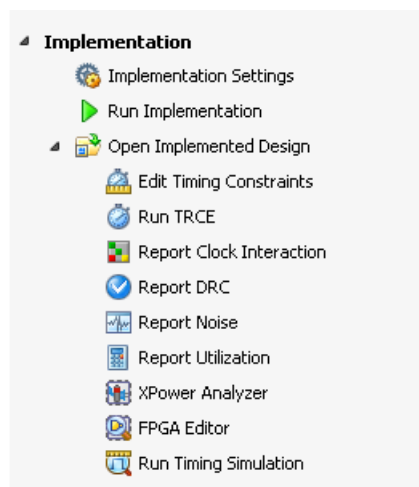


図 1-2 : Flow Navigator : [Implementation] セクション

Vivado デザイン ツールでは、Tcl API も提供されています。Tcl API を使用すると、すべてのデザイン フローをスクリプトで実行でき、デザイン フローを要件に合わせてカスタマイズできます。

# インプリメンテーションの前に

Vivado Design Suite ではさまざまなデザイン フローが提供されており、多種のデザイン ソースがサポートされています。FPGA にダウンロード可能なビットストリームを生成するには、インプリメンテーションを実行する必要があります。インプリメンテーションは、論理ネットリストをターゲット ザイリンクス デバイスの物理的なアレイにマップするための一連の手順を指します。これらの手順には、ロジック最適化、ロジック セルの配置、セル間の接続の配線が含まれます。次のセクションでは、Vivado Design Suite でデザインにインプリメンテーションを実行するために必要な情報を示します。

## プロジェクト モードと非プロジェクトモード

Vivado ツールでは、プロジェクト ファイル (.xpr) とディレクトリ構造を作成して、デザイン ソース ファイルの管理、合成およびインプリメンテーション結果の保存、デザイン フローのプロジェクト ステータスの管理を実行できます。または、プロジェクト ファイルおよびローカル ディレクトリを作成せずに、すべてメモリ内で作業することも可能です。

プロジェクト モード (プロジェクト ベース デザイン) では、ディスク上にディレクトリ構造が作成され、それを利用してデザイン ソース、run の結果、およびプロジェクト ステータスが管理されます。デザイン データ、プロセス、およびステータスを自動管理するには、Vivado プロジェクト ファイル (.xpr) に保存されるプロジェクト インフラストラクチャが必要です。また、デザイン フローの主要な段階のチェックポイント ファイルが、自動的にローカル プロジェクト ディレクトリに保存されます。

プロジェクト ファイルを使用しないコンパイル形式のフローは、非プロジェクト モードまたは非プロジェクト バッチ フローと呼ばれます。非プロジェクト モードの利点は、メモリ内でデザインに作業を実行することが可能であることです。ソース ファイルおよびデザイン制約が現在の場所からメモリに読み込まれ、中間ファイルを作成せずにメモリ内のデザインに対してデザイン フローが実行されます。非プロジェクト モードでは、Tcl コマンドを使用し、デザイン パラメーターおよびインプリメンテーション オプションを設定して、各手順を個別に実行する必要があります。変更を保存したり手順を再実行したりする必要なく、デザインに変更を加え、デザイン フローを続行できます。デザイン フローのどの段階でも、レポートを生成したり、デザイン チェックポイント (.dcp) を保存したりできます。



**重要:** 非プロジェクト モードでは、Vivado デザイン ツールを終了すると、メモリ内のデザインは失われます。そのため、合成、配置、配線などの主な手順が終了したら、デザイン チェックポイントを保存するようにします。

デザイン チェックポイントは、プロジェクト モードおよび非プロジェクト モード両方で保存できます。デザイン チェックポイントを読み込むことができるのは、非プロジェクト モードのみです。

Vivado Design Suite のプロジェクト モードと非プロジェクト モードには、さまざまな違いがあります。Vivado インプリメンテーションは、プロジェクト ベースのデザインおよび非プロジェクト ベースのデザインのどちらでも実行できます。Vivado IDE および Tcl API も、両方のフローで使用できます。ただし、Vivado IDE および Vivado インプリメンテーションの一部の機能は、非プロジェクト モードでは使用できません。その例を次に示します。<sup>(1)</sup>

- Flow Navigator
- デザイン ステータス インジケータ
- IP 統合
- インプリメンテーション run および run ストラテジ
- [Design Runs] ビュー
- [Messages] ビュー
- [Reports] ビュー

1. このリストには、非プロジェクト モードでサポートされない機能すべては含まれていません。完全なリストではありません。

非プロジェクト ベースのデザインをインプリメントするには、Tcl コマンド `opt_design`、`place_design`、および `route_design` を個別に実行する必要があります。インプリメンテーションの各段階は、Vivado IDE または Tcl コンソールから対話的に実行するか、カスタム Tcl スクリプトを使用して実行できます。デザイン フローは、必要に応じてレポート コマンドや最適化を追加してカスタマイズすることも可能です。詳細は、[11 ページの「非プロジェクト モードでのインプリメンテーションの実行」](#)を参照してください。

このガイドでは、プロジェクト モードおよび非プロジェクト モードでのインプリメンテーションの実行について詳細に説明します。プロジェクト モードまたは非プロジェクト モードを使用した Vivado Design Suite の実行に関する詳細は、『Vivado Design Suite ユーザー ガイド：デザイン フローの概要』(UG892) および『Vivado Design Suite ユーザー ガイド：Vivado IDE の使用』(UG893) を参照してください。

## RTL および合成済みデザイン

Vivado Design Suite を使用して、RTL 開発、IP のカスタマイズ、合成、インプリメンテーション、デバイスのプログラムおよび検証まで、FPGA デザイン フロープロセスすべてを管理できます。Verilog、SystemVerilog、VHDL ファイルなどの HDL ソースを追加できます。ザイリンクス IP カタログから、あらかじめ定義されコンフィギュレーションされた IP コアを追加できます。System Generator からの DSP デジタル信号処理モジュール、Vivado 高位合成 (HLS) からの C ベースの DSP モジュール、Xilinx Platform Studio (XPS) からのエンベデッド プロセッサ モジュールも追加できます。

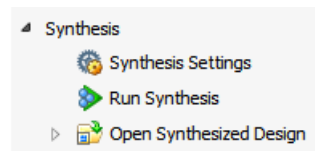
Vivado Design Suite では、ネットリスト デザインもサポートされており、ザイリンクス ツールまたはサードパーティ ツールで合成済みのネットリストをインポートできます。ネットリストの入力フォーマットは、構造 Verilog、SystemVerilog、EDIF、またはザイリンクス NGC です。Vivado Design Suite でサポートされるソース ファイルおよびプロジェクト タイプの詳細は、『Vivado Design Suite ユーザー ガイド：システム デザイン入力』(UG895) を参照してください。

Vivado インプリメンテーションを実行するには、合成済みネットリストが必要です。デザインは合成済みネットリストから、または RTL ソース ファイルから開始できます。RTL ソースから開始する場合は、インプリメンテーションの前に Vivado 合成または XST を実行する必要があります。Vivado IDE では、これが自動的に管理されます。合成されていないデザインをインプリメントしようとする、まず合成を実行するかどうかを選択するオプションが表示されます。Vivado 合成の詳細は、『Vivado Design Suite ユーザー ガイド：合成』(UG901) を参照してください。

非プロジェクト モードでは、Tcl コマンド `synth_design` を使用して、合成を実行し、合成済みデザインを開く必要があります。デザインを開くには、Tcl コマンド `link_design` も使用できます。詳細は、[第 2 章の「合成済みデザインを開く」](#)を参照してください。

プロジェクト モードでは、RTL デザインの合成後、またはネットリスト ベース デザインを開いて、インプリメンテーション前のデザイン ネットリストを読み込むことができます。Vivado IDE で合成済みデザインを開くには、次のいずれかの方法を使用します。

- メイン メニューから [Flow] → [Open Synthesized Design] をクリックします。
- Flow Navigator で [Synthesis] → [Open Synthesized Design] をクリックします。
- [Design Runs] ビューで run を右クリックし、[Open Synthesized Design] をクリックします。





## IP 中心のデザイン

Vivado IP カタログを使用すると、IP を設定、インプリメント、および検証できます。IP は、スタンドアロンのモジュールとして、またはシステム レベル デザインの一部として設定および検証できます。

IP カタログには、ザイリンクス LogicCORE™ IP すべてと、カタログに追加されたユーザー定義の IP およびサードパーティ IP が表示されます。カタログには、各 IP のタイプ、バージョン、データシート、ライセンスなどの情報が含まれています。RTL デザインに IP コアを追加するには、インスタンス化テンプレートをシステム レベル デザインに挿入します。IP は、ネットリストではなく RTL ソースとして作成されます。合成およびインプリメンテーションを実行すると、デザインのほかの部分と共に IP が合成およびインプリメントされます。IP は、スタンドアロン モジュールとして合成し、そのネットリストをネットリスト デザインに追加することも可能です。サポートされる IP ネットリスト フォーマットは、ザイリンクスのフォーマット (.xco、.xci、.ngc)、Verilog (.v)、および EDIF (.edf) です。Vivado ツールでの IP デザインのサポートについては、『Vivado Design Suite ユーザー ガイド : IP を使用した設計』(UG896) を参照してください。

## 制約

インプリメンテーションを実行する際は、合成済みネットリストと共にデザイン制約を供給する必要があります。制約セットは、XDC で記述されたデザイン制約を含む複数の制約ファイルのセットで、これらをデザインに適用できます。デザイン制約には、2 種類あります。

- 物理制約 : ピン配置、BRAM、LUT、フリップフロップなどのセルの絶対配置または相対配置、およびデバイスのコンフィギュレーション設定を定義します。
- タイミング制約 : 業界標準の SDC で記述し、デザインの周波数要件を定義します。タイミング制約を設定しない場合、デザインがワイヤの長さおよび配線の密集度により最適化されます。

**注記 :** タイミング制約を設定しない場合、Vivado インプリメンテーションでデザインのパフォーマンスを評価したり、向上するための処理は実行されません。



**重要 :** Vivado Design Suite では、UCF フォーマットはサポートされません。UCF 制約を XDC コマンドに移行する方法は、『Vivado Design Suite 移行手法ガイド』(UG912) を参照してください。

制約セットには、複数の制約ファイルを含めることができます。異なる物理制約ファイルおよびタイミング制約ファイルを含む複数の制約セットを作成できます。マスター制約ファイルを用意し、デザインへの変更は新しい制約ファイルに保存するなど、さまざまな利用方法が可能です。制約を機能に応じて別の制約ファイルに分けておくと、制約ストラテジ全体がわかりやすくなり、タイミングおよびインプリメンテーションを変更しやすくなります。

1 つのプロジェクトに対して複数の制約セットを作成し、異なるインプリメンテーション run で異なる制約セットをアクティブにし、異なる設定をテストできます。また、合成とインプリメンテーションに異なる制約セットを指定できます。デザイン要件が満たされるよう、合成、シミュレーション、インプリメンテーションで異なる制約を適用してみることが可能です。デザイン制約を複数の制約セットを使用して整理すると、次のような利点があります。

- 同じプロジェクトで異なるザイリンクス FPGA をターゲットとして設定できます。ターゲット パーツが異なると、物理制約およびタイミング制約も異なるものにする必要がある場合があります。
- さまざまな条件でデザインを実行できます。制約セットを使用して、異なるフロアプランを適用したり、デザインの制約を厳しくしたりできます。
- 制約の変更を管理しやすくなります。マスター制約の代わりに、別の制約ファイルに保存した制約を使用できます。



**ヒント :** タイミング制約を検証するには、合成済みデザインに report\_timing\_summary コマンドを使用するのがよい方法です。問題の発生しやすい制約は、インプリメンテーションの前に修正してください。

制約の定義および制約での作業の詳細は、『Vivado Design Suite ユーザー ガイド : 制約の使用』(UG903) を参照してください。

## デザイン チェックポイント

インプリメンテーションプロセスでは、デザイン ネットリストおよび制約だけでなく、配置配線データを保存するのに物理デザイン データベースも使用されます。Vivado ツールでは、デザイン フローの主要な段階でこの物理データベースを保存および復元するためのメカニズムとして、デザイン チェックポイントが使用されます。チェックポイントは、フローの特定の地点におけるデザインのスナップショットです。

現在のネットリスト、インプリメンテーション中に実行された最適化、およびインプリメンテーション結果が、デザイン チェックポイント ファイルに保存されます。チェックポイント デザインに対しては、Tcl コマンドを使用してデザイン フローの残りの段階を実行できますが、新しいデザイン ソースで変更を加えることはできません。

Vivado IDE では、次の方法を使用してチェックポイントの保存および読み込みを実行します。

- チェックポイントの保存 : [File] → [Write Checkpoint] をクリックします。フローの任意の段階におけるデザイン データベースのスナップショットが保存されます。これにより、拡張子が .dcp のファイルが作成されます。これに相当する Tcl コマンドは `write_checkpoint` です。
- チェックポイントの読み込み : [File] → [Open Checkpoint] をクリックします。Vivado Design Suite で指定のチェックポイントを開きます。デザイン チェックポイントは、Vivado デザイン ツールで別のプロジェクトとして開き、既存のプロジェクトに読み込むことはできません。これに相当する Tcl コマンドは `read_checkpoint` です。

# 非プロジェクト モードでのインプリメンテーションの実行

合成済みデザインまたはネットリストをターゲットのザイリンクス FPGA にインプリメントするには、ソース ファイルとデザイン制約に対してインプリメンテーションと呼ばれる最適化、配置、配線を含む複数プロセスを実行する必要があります。

非プロジェクト モードでは、複数の Tcl コマンドを順に実行するか、デザイン フロー全体を定義した Tcl スクリプトを使用します。Tcl コマンドは、Vivado IDE の Tcl コンソールから入力するか、Vivado Design Suite Tcl シェルの Tcl プロンプトから入力します。

## 非プロジェクト モードのサンプル スクリプト

次に、非プロジェクト モードでインプリメンテーションを実行するスクリプト例を示します。

```
# Step 1:Read in top-level EDIF netlist from synthesis tool
read_edif c:/top.edf
# Read in lower level IP core netlists
read_edif c:/core1.edf
read_edif c:/core2.edf

# Step 2:Specify target device and link the netlists
# Merge lower level cores with top level into single design
link_design -part xc7k325tfbg900-1 -top top.edf

# Step 3:Read XDC constraints to specify timing requirements
read_xdc c:/top_timing.xdc
# Read XDC constraints that specify physical constraints such as pin locations
read_xdc c:/top_physical.xdc

# Step 4:Optimize the design with default settings
opt_design

# Step 5:Place the design with effort level set to high
place_design -effort_level high

# Step 6:Route the design with effort level set to high
route_design -effort_level high

# Step 7:Run Timing Summary Report to see timing results
report_timing_summary -file post_route_timing.rpt
# Run Utilization Report for device resource utilization
report_utilization -file post_route_utilization.rpt

# Step 8:Write checkpoint to capture the design database;
# The checkpoint can be used for design analysis in Vivado IDE or TCL API
write_checkpoint post_route.dcp
```

## サンプル スクリプトの詳細

上記のサンプル スクリプトは、次の段階から構成されています。

### 1. デザイン ソース ファイルの読み込み

このサンプルでは、デザイン ソースは EDIF ネットリスト ファイルです。非プロジェクト モードでは、RTL デザイン フローもサポートされています。その場合、ソース ファイルを読み込んでインプリメンテーションの前に合成を実行します。

`read_* Tcl` コマンドは、非プロジェクト モードで使用し、**Vivado Design Suite** でディスク上のファイルを読み込んでメモリ内にデザインを構築します。ファイルがコピーされたり、ファイルの依存関係が作成されることはありません。そのため、非プロジェクト モードは非常に柔軟ですが、ユーザーがソース デザイン ファイルの変更を監視し、それに応じてデザインをアップデートする必要があります。

### 2. メモリ内へのデザインの構築

このサンプル スクリプトでは、`link_design` コマンドを使用してデザインのメモリ内表示を構築します。このコマンドは、ネットリスト ベースのソース ファイルをツールに読み込み、ザイリンクスのデバイス情報と結合して、メモリ内にデザイン データベースを作成します。非プロジェクト モードでのすべての操作は、**Vivado** ツール内のインメモリ データベースに対して実行されます。

**Vivado** ツールをバッチ モードで実行している場合でも、**Tcl** シェル モードで対話的に `Tcl` コマンドを実行している場合でも、グラフィカル モードでデザイン データを **Vivado IDE** で表示している場合でも、メモリ内のデザインは **Vivado** ツール内に存在します。

### 3. デザイン制約の読み込み

**Vivado Design Suite** では、[9 ページの「制約」](#)に説明するように、デザイン制約を使用してデザインの物理的特性およびタイミング特性を定義します。`read_xdc` コマンドは、XDC 制約ファイルを読み込み、メモリ内のデザインに適用します。



**ヒント** : プロジェクト モードでは、異なる目的で複数の制約ファイルを含む制約セットを定義できますが、非プロジェクト モードでは、同じ操作に複数の `read_xdc` コマンドを使用します。

### 4. ロジック最適化の実行

配置配線の準備としてロジック最適化を実行します。最適化の目的は、ロジック デザインを、ターゲット パーツの物理リソースに配置する前に簡略化することです。**Vivado** のロジック最適化では、デザイン要件を満たすため、さまざまな最適化機能が提供されています。詳細は、[37 ページの「ロジック最適化」](#)を参照してください。

### 5. 基本ロジック エレメントの配置

デザインの全体的な配置を実行します。デザインの階層および配置の困難さによっては、配置が複数の段階で実行されることもあります。詳細は、[39 ページの「配置」](#)を参照してください。

### 6. デザインの配線

デザインの必要な配線を完成させます。**Vivado** 配線では、すべての場合に通常配線が使用され、困難なデザインには再配線を使用して詳細な制御が可能です。詳細は、[42 ページの「配線」](#)を参照してください。

### 7. レポートの生成

**Vivado Design Suite** で提供される多数のレポートのうち 2 つを生成します。非プロジェクト モードでは、適切な `Tcl` コマンドを使用して、各レポートを指定する必要があります。レポートはファイルに出力するか、**Vivado IDE** に表示して確認できます。詳細は、[30 ページの「インプリメンテーション レポートの表示」](#)を参照してください。

## 8. デザイン チェックポイントの保存および Vivado ツールの終了

メモリ内のデザイン (最適化されたネットリスト、物理制約およびタイミング制約、ザイリンクス パーツ情報、配置配線情報) をデザイン チェックポイント ファイルに保存します。非プロジェクト モードでは、デザイン チェックポイント ファイルを保存することで、後でデザインを読み込んで解析したり変更したりできます。詳細は、10 ページの「デザイン チェックポイント」を参照してください。

# プロジェクト モードでのインプリメンテーションの実行

プロジェクト モードでは、特定の合成結果およびデザイン制約を使用するよう設定したインプリメンテーション run を定義でき、1 つのデザインに対して複数のストラテジを実行できます。また、特定のデザイン要件を満たすためにインプリメンテーション ストラテジをカスタマイズし、それらをほかのデザインでも使用できるように保存できます。



**重要:** 非プロジェクト モードでは、あらかじめ定義されたインプリメンテーション run およびストラテジはサポートされません。Tcl コマンドを使用して、インプリメンテーションプロセスの各段階を手動で実行する必要があります。詳細は、11 ページの「非プロジェクト モードでのインプリメンテーションの実行」を参照してください。

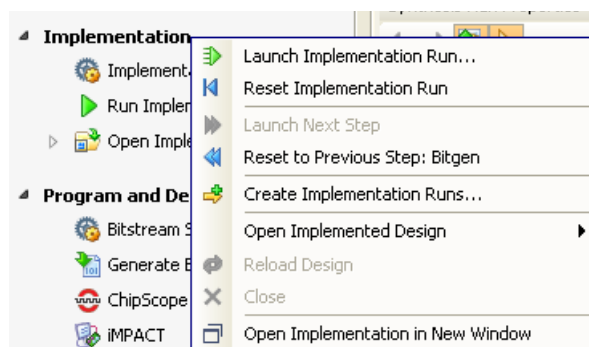
## インプリメンテーション run の作成

新しいインプリメンテーション run を作成して起動し、最適な結果が得られるようさまざまなオプションを試すことができます。各 run は順次起動するか、複数のローカル CPU で同時に起動できます。Linux システムでは、リモートサーバーで run を実行することも可能です。詳細は、付録 A 「リモート ホストの使用」を参照してください。

インプリメンテーション run を定義するには、次のいずれかの方法を使用します。

- [Flow] → [Create Runs] をクリックします。
- Flow Navigator で [Implementation] を右クリックし、[Create Implementation Runs] をクリックします。
- [Design Runs] ビューを右クリックし、[Create Runs] をクリックします。

Create New Runs ウィザードが開きます。ウィザードの最初のページは、コマンドのサマリです。[Next] をクリックします。



**注記:** [Flow] → [Create Runs] をクリックした場合、Create New Runs ウィザードの最初のページで [Implementation] をオンにします。

図 1-3 に示す [Set-Up Implementation Runs] ページが表示されます。

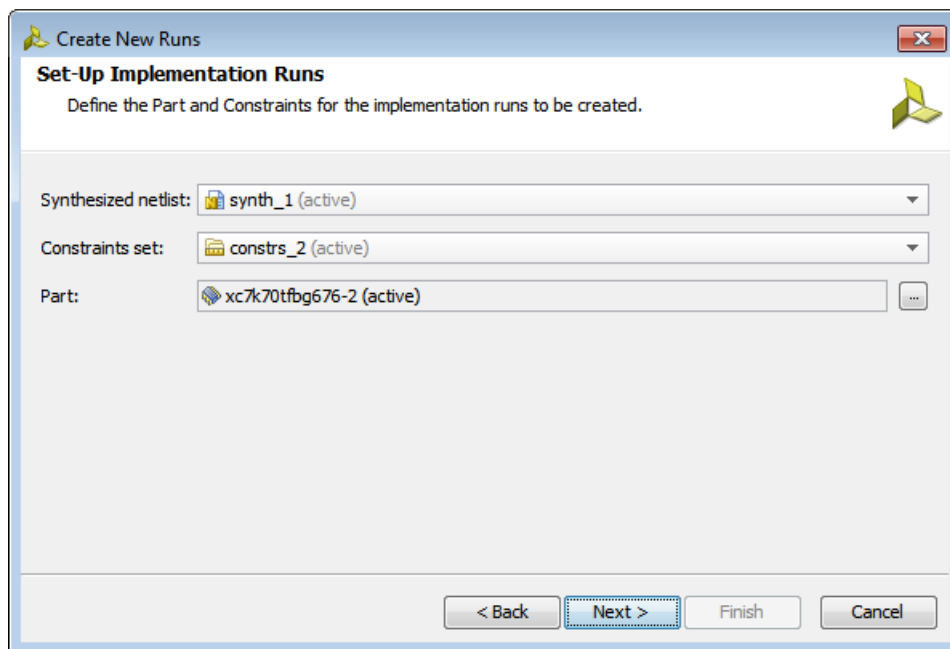


図 1-3 : Create New Runs ウィザード : [Set-Up Implementation Runs] ページ

1. [Synthesized netlist] で合成済みネットリストを選択します。

選択した合成 run が実行されていない場合は実行され、その合成済みネットリストがインプリメントに使用されます。または、サードパーティ合成ツールからプロジェクトにインポートされた合成済みネットリストも選択できます。詳細は、『Vivado Design Suite ユーザー ガイド : 合成』(UG901) を参照してください。

デフォルトは、[Design Runs] ビューで現在アクティブな run です。[Design Runs] ビューについては、[17 ページの「\[Design Runs\] ビューの使用」](#)を参照してください。

2. [Constraints set] で制約セットを選択します。

選択した制約セットがインプリメンテーションで適用されます。最適化、配置、および配線は、指定された制約セットの物理制約およびタイミング制約に基づいて実行されます。詳細は、『Vivado Design Suite ユーザー ガイド : 制約の使用』(UG903) を参照してください。

3. [Part] でターゲット パーツを選択します。

デフォルトでは、制約セットおよびターゲット パーツは、[Create New Runs] コマンド実行したときのプロジェクト設定に基づいて定義されます。プロジェクト設定については、『Vivado Design Suite ユーザー ガイド : Vivado IDE の使用』(UG893) を参照してください。

異なる制約セットまたはターゲット パーツを使用する run を作成するには、[Create New Runs] コマンドを使用します。これらの値は、[Design Runs] ビューで run を選択し、[Run Properties] ビューで変更できます。詳細は、[17 ページの「インプリメンテーション run 設定の変更」](#)を参照してください。

4. [Next] をクリックします。

図 1-4 に示す [Choose Implementation Strategies] ページが表示されます。

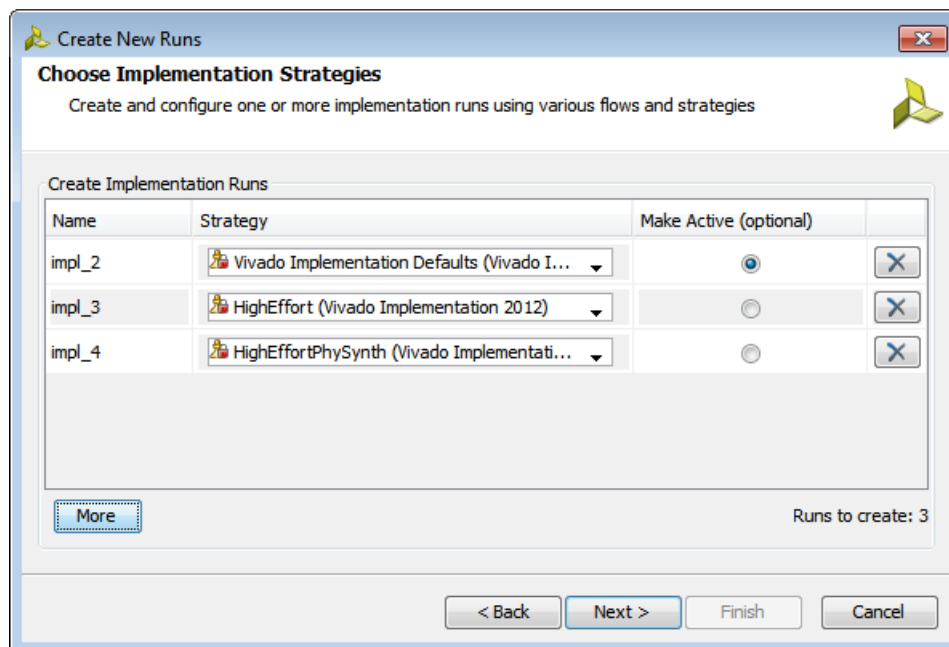


図 1-4 : [Choose Implementation Strategies] ページ

5. [Name] 列に run の名前を入力するか、デフォルトのままにします。
6. [Strategy] 列のドロップダウン リストから、新しい run のストラテジを選択します。

ストラテジとは、インプリメンテーション結果を制御する Vivado インプリメンテーション機能オプションを定義した設定のことです。Vivado Design Suite では、定義済みのストラテジが提供されています。また、独自のインプリメンテーション ストラテジを作成することも可能です。詳細は、21 ページの「ストラテジの定義」を参照してください。

- [Vivado Implementation Defaults]: 適度なランタイムでタイミング クロージャが満たされるように試行します。
- [HighEffort]: タイミング クロージャを満たすことに焦点が置かれ、ランタイムは増加します。
- [HighEffortPhySynth]: タイミング クロージャを満たすことに焦点が置かれ、ランタイムは増加します。配置後に物理合成がイネーブルになります。
- [LowEffort]: 配置配線が低いエフォート レベルで実行されます。初期のインプリメンテーション試行で有益です。
- [QuickEffort]: 非タイミングドリブンの配置配線が実行されます。すべてのタイミング制約が無視されます。制約されていないデザインの配置配線性を評価するためにのみ使用してください。

run を実行する前に、インプリメンテーションプロセスの各段階の設定を、選択したストラテジのデフォルト設定から変更できます。変更した設定を新しいストラテジとしても保存できます。詳細は、17 ページの「インプリメンテーション run 設定の変更」を参照してください。

7. オプションで [Make Active] をオンにし、新しい run をアクティブ run にします。新規 run を複数作成する場合、アクティブにできる run は 1 つだけです。Vivado IDE には、アクティブな run の実行結果が表示されます。
8. [More] ボタンをクリックし、追加の run を定義します。追加 run の名前とストラテジを指定します (図 1-4)。



9. [Next] をクリックします。

図 1-5 に示す [Launch Options] ページが開きます。

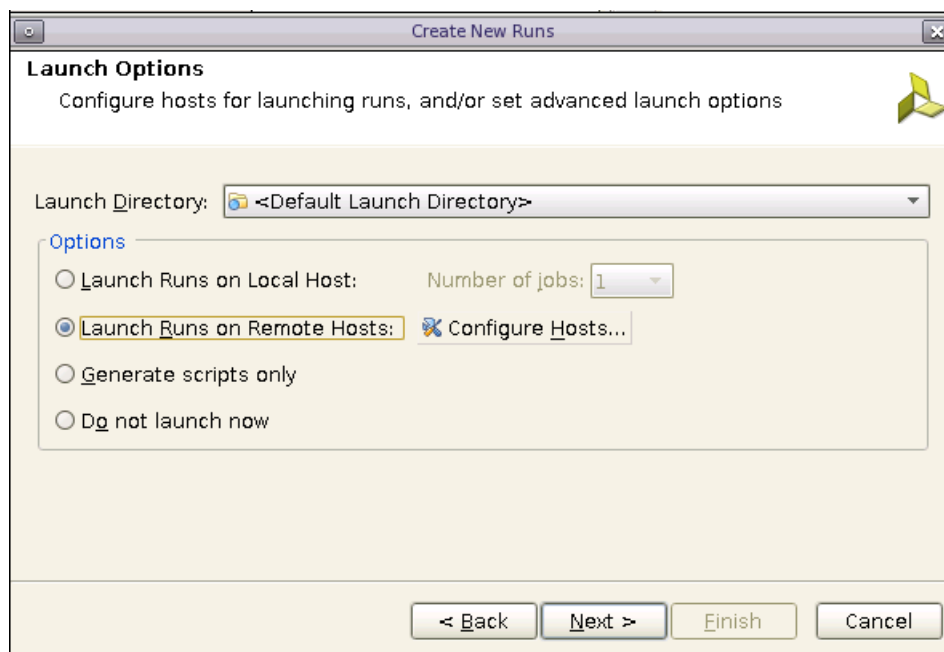


図 1-5 : [Specify Launch Options] ダイアログ ボックス

10. [Launch Directory] でインプリメンテーション run のデータを作成して保存するディレクトリを指定します。

デフォルトのディレクトリは、ローカルプロジェクト ディレクトリ構造に含まれます。デフォルトでは、インプリメンテーション run のファイルは次のディレクトリに保存されます。

`<project_name>/<project_name>.runs/<run_name>`



**ヒント** : プロジェクト ファイルには絶対パスが記述されるので、プロジェクト ディレクトリ外の場所を指定すると、プロジェクトを移動しにくくなります。

11. 実行オプションを指定します。

- [Launch Runs on Local Host] : run をローカル マシンで実行します。
  - [Number of jobs] : 複数の run を同時実行する際に使用するローカルプロセッサの数を指定します。各 run が各プロセッサで実行されます。Vivado では、インプリメンテーションのマルチスレッドはサポートされません。
- [Launch Runs on Remote Hosts] (Linux のみ) : リモート ホストを使用してジョブを実行します。詳細は、付録 A 「リモート ホストの使用」を参照してください。
  - [Configure Hosts] : リモート ホストを設定します。
- [Generate scripts only] : run を実行せずに、run ディレクトリおよび run スクリプトをエクスポートおよび作成します。スクリプトは、Vivado IDE ツールの環境外で後で実行できます。
- [Do not launch now] : 新しい run を保存しますが、run を実行または run スクリプトを作成しません。

12. [Next] をクリックし、[Create New Runs Summary] ページを確認します。

13. [Finish] をクリックすると、定義した run が作成され、指定の実行オプションが実行されます。

新しい run が [Design Runs] ビューに追加されます。

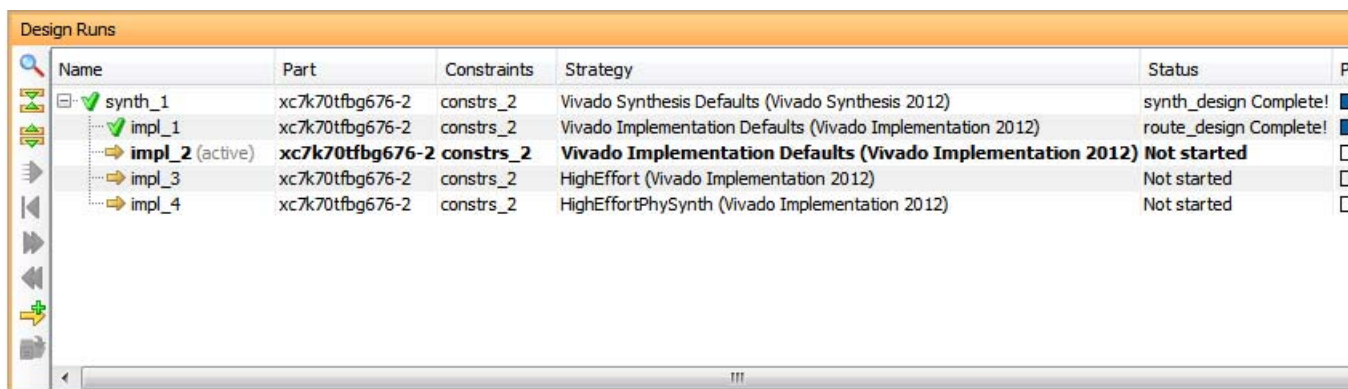


## [Design Runs] ビューの使用

[Design Runs] ビューには、プロジェクトで作成された合成 run とインプリメンテーション run のすべてが表示され、それらを設定、管理、実行するためのコマンドも表示されます。

[Design Runs] ビューが表示されていない場合は、[Window] → [Design Runs] をクリックして表示します。図 1-6 に、[Design Runs] ビューを示します。

各インプリメンテーション run は、合成 run の下の階層にインデントされて表示されます。1 つの合成 run に、複数のインプリメンテーション run を含めることができます。プラス記号 (+) やマイナス記号 (-) をクリックすると、合成 run のツリー表示を展開したり、閉じたりできます。[Design Runs] ビューは、ツリー形式の表で示されます。このビューのデータを列を使用して並べ替える方法は、『Vivado Design Suite ユーザー ガイド : Vivado IDE の使用』(UG893) を参照してください。



Name	Part	Constraints	Strategy	Status
synth_1	xc7k70tfbg676-2	constrs_2	Vivado Synthesis Defaults (Vivado Synthesis 2012)	synth_design Complete!
impl_1	xc7k70tfbg676-2	constrs_2	Vivado Implementation Defaults (Vivado Implementation 2012)	route_design Complete!
impl_2 (active)	xc7k70tfbg676-2	constrs_2	<b>Vivado Implementation Defaults (Vivado Implementation 2012)</b>	<b>Not started</b>
impl_3	xc7k70tfbg676-2	constrs_2	HighEffort (Vivado Implementation 2012)	Not started
impl_4	xc7k70tfbg676-2	constrs_2	HighEffortPhySynth (Vivado Implementation 2012)	Not started

図 1-6 : [Design Runs] ビュー

[Design Runs] ビューには、run のステータスが実行されていないか、進行中か、完了したか、最新の状態でないかが示されます。ソース ファイル、制約、またはプロジェクト設定を変更すると、run は最新の状態ではなくなります。[Design Runs] ビューでは、run をリセットしたり、古い run のデータを削除したりできます。

Vivado IDE でアクティブにできるのは、1 つの合成 run と 1 つのインプリメンテーション run のみです。Vivado IDE のほかのビューには、アクティブな run の情報が表示されます。[Log] ビューおよび [Reports] ビュー、ステータスバー、[Project Summary] ビューには、アクティブな run の情報が表示されます。[Project Summary] ビューには、アクティブな run のコンパイル、リソース、およびサマリ情報が表示されます。

アクティブな run は、[Design Runs] ビューに太字で示されます。別の run をアクティブにするには、[Design Runs] ビューでその run を右クリックし、ポップアップ メニューから [Make Active] コマンドをクリックします。

## インプリメンテーション run 設定の変更

[Design Runs] ビューで run を選択すると、図 1-7 に示すように、[Run Properties] ビューにその run の現在の設定が表示されます。[Run Properties] ビューでは、次のオプションを変更できます。

- [Name] : run の名前
- [Part] : ターゲット パーツ
- [Description] : run の説明
- [Constraints] : インプリメンテーションで使用し、新しい制約を保存する制約セット

[Run Properties] ビューの詳細は、『Vivado Design Suite ユーザー ガイド : Vivado IDE の使用』(UG893) を参照してください。

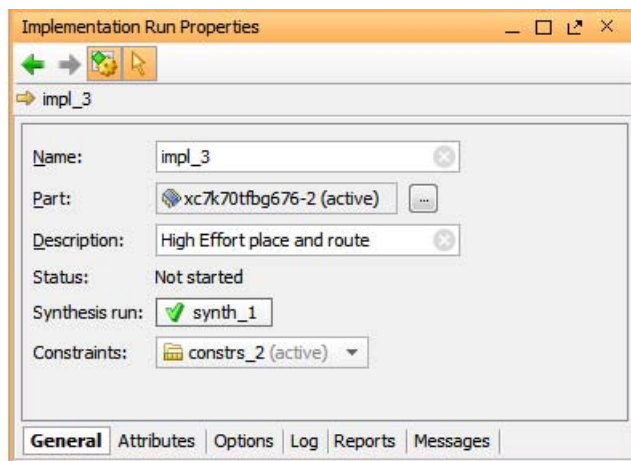


図 1-7 : [Implementation Run Properties] ビュー

Vivado インプリメンテーション機能で使用するオプションも変更できます。[Design Runs] ビューで run を右クリックし、[Change Run Settings] をクリックして、[Design Run Settings] ダイアログ ボックスを開きます (図 1-8)。



**ヒント** : 設定の変更は、run のステータスが「Not started」の場合にのみ可能です。run を右クリックして [Reset Runs] をクリックすると、run のステータスを「Not started」に戻すことができます。詳細は、20 ページの「[run のリセットおよび削除](#)」を参照してください。

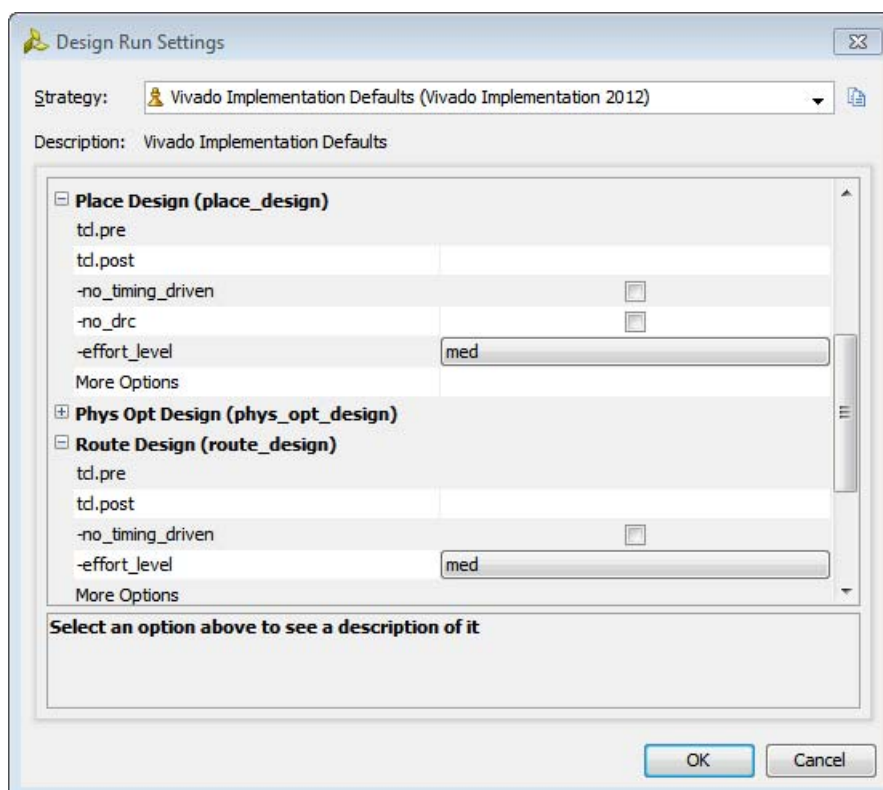


図 1-8 : [Design Run Settings] ダイアログ ボックス

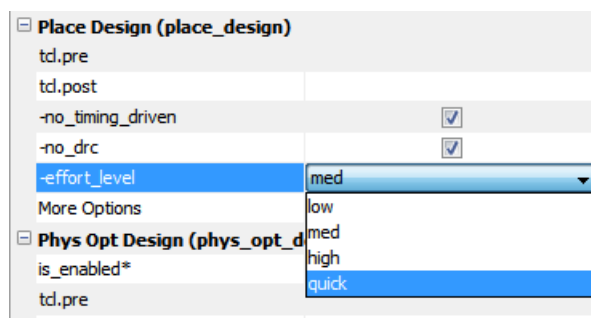
[Design Run Settings] ダイアログ ボックスには、run に適用されているインプリメンテーション ストラテジ、インプリメンテーション プロセスの各段階でそのストラテジに関連するコマンド オプションが表示されます。

- [Strategy]: インプリメンテーション run に適用するストラテジを選択します。Vivado Design Suite では、定義済みのストラテジが提供されています。また、独自のインプリメンテーション ストラテジを作成することも可能です。詳細は、[21 ページの「ストラテジの定義」](#)を参照してください。
- [Description]: 選択したインプリメンテーション ストラテジの説明を表示します。
- [Options]: Vivado インプリメンテーション プロセスの各段階 (opt\_design、power\_opt\_design、place\_design、phys\_opt\_design、route\_design、および write\_bitstream) のコマンド ライン オプションを表示します。

特定のコマンド オプションをクリックすると、そのオプションの簡単な説明が下に表示されます。各インプリメンテーション段階の詳細と設定可能なオプションは、[第 2 章「インプリメンテーション コマンド」](#)を参照してください。

コマンド オプションを変更するには、そのコマンド オプションの右側をクリックします。

- 定義済みの値から選択するオプションは、ドロップダウン リストから選択します。
- イネーブル / ディスエーブルにするオプションは、チェック ボックスのオン / オフを切り替えます。
- ユーザー定義の値を指定できるオプションは、値を入力します。
- ファイル名およびパスを指定するオプションは、ファイルを選択するダイアログ ボックスが開き、ファイルを選択できます。
- 各段階の最初 (tcl.pre) と最後 (tcl.post) に Tcl スクリプトも追加でき、インプリメンテーションの前後に特定のタスクを実行できます。たとえば、デザインの配置前後にタイミング レポートを生成して、タイミング結果を比較できます。
- [Save Design As]: [Strategy] フィールドの右側にあるボタンで、ストラテジへの変更を新しいストラテジとして保存し、今後使用できるようにします。このコマンドを使用しない場合、変更は現在のインプリメンテーション run には保存されますが、今後使用することはできません。



## run ステータスの理解

Vivado IDE では、run のステータスによって、run を処理してインプリメンテーションを開始します。ステータスは、[Design Runs] ビューに表示されます ([17 ページの図 1-6](#))。run のステータスに応じて、次のように処理されます。

- [Not started]: run がすぐに実行されます。
- [Error]: まず run がリセットされ、不完全な run データが削除されてから、run が再開されます。
- [Complete] (または [Out-of-Date]): run をリセットするかどうかを確認するメッセージが表示されます。

## run のリセットおよび削除

run をリセットするには、[Design Runs] ビューで run を右クリックし、ポップアップ メニューから [Reset Runs] コマンドをクリックします。インプリメンテーション run をリセットすると、インプリメンテーションの最初の段階 (opt\_design) に戻ります。

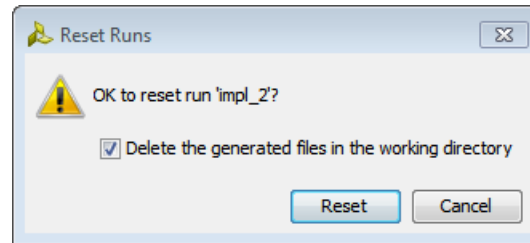


図 1-9 : [Reset Runs] ダイアログ ボックス

[Reset Runs] コマンドを確認するメッセージと、run ディレクトリから生成されたファイルを削除するオプションが表示されます。デフォルトでは、生成されたファイルは削除されます。生成された run ファイルを保存する場合は、このオプションをオフにします。

[Design Runs] ビューで run を右クリックし、[Delete] をクリックすると、run を削除できます。[Delete Runs] コマンドを確認するメッセージと、run ディレクトリから生成されたファイルを削除するオプションが表示されます。このオプションは、デフォルトではオンになっています。

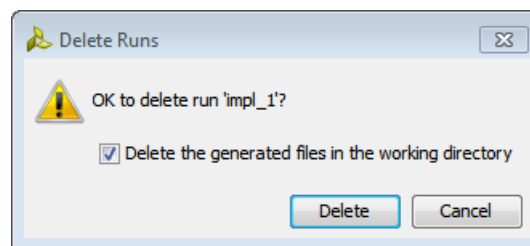


図 1-10 : [Delete Runs] ダイアログ ボックス

## インプリメンテーション ストラテジのカスタマイズ

新しいインプリメンテーション run を定義すると、デフォルトのインプリメンテーション設定が使用されます。これらの設定は変更できます。

21 ページの図 1-11 に、[Project Settings] ダイアログ ボックスの [Implementation] ページを示します。このダイアログ ボックスは、メイン メニューから [Tools] → [Project Settings] をクリックすると開きます。

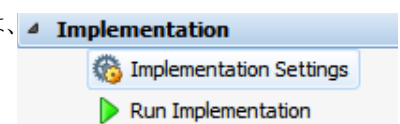


**ヒント**：非プロジェクト モードを使用している場合は、[Project Settings] コマンドは使用できません。インプリメンテーション ストラテジをバッチ モードで使用可能な Tcl スクリプトとして定義して保存するか、Vivado IDE で対話的に定義します。

アクティブなインプリメンテーション run のインプリメンテーション設定は、Flow Navigator で [Implemented Settings] をクリックしても開くことができます。

次のインプリメンテーション設定を指定できます。

- [Default Constraint Set]：インプリメンテーション run でデフォルトで使用する制約セットを選択します。
- [Strategy]：インプリメンテーション run に適用するストラテジを選択します。Vivado Design Suite では、定義済みのストラテジが提供されています。また、独自のインプリメンテーション ストラテジを作成することも可能です。詳細は、21 ページの「ストラテジの定義」を参照してください。



- [Save Design As] : [Strategy] フィールドの右側にあるボタンで、ストラテジへの変更を新しいストラテジとして保存し、今後使用できるようにします。
- [Description] : 選択したインプリメンテーション ストラテジの説明を表示します。



Vivado ツールの標準インプリメンテーション ストラテジの説明は変更できませんが、標準ストラテジを変更すると、新しいストラテジとして保存でき、そのときに説明も変更できます。

ユーザー定義のストラテジの説明は、新しい説明を入力して変更できます。

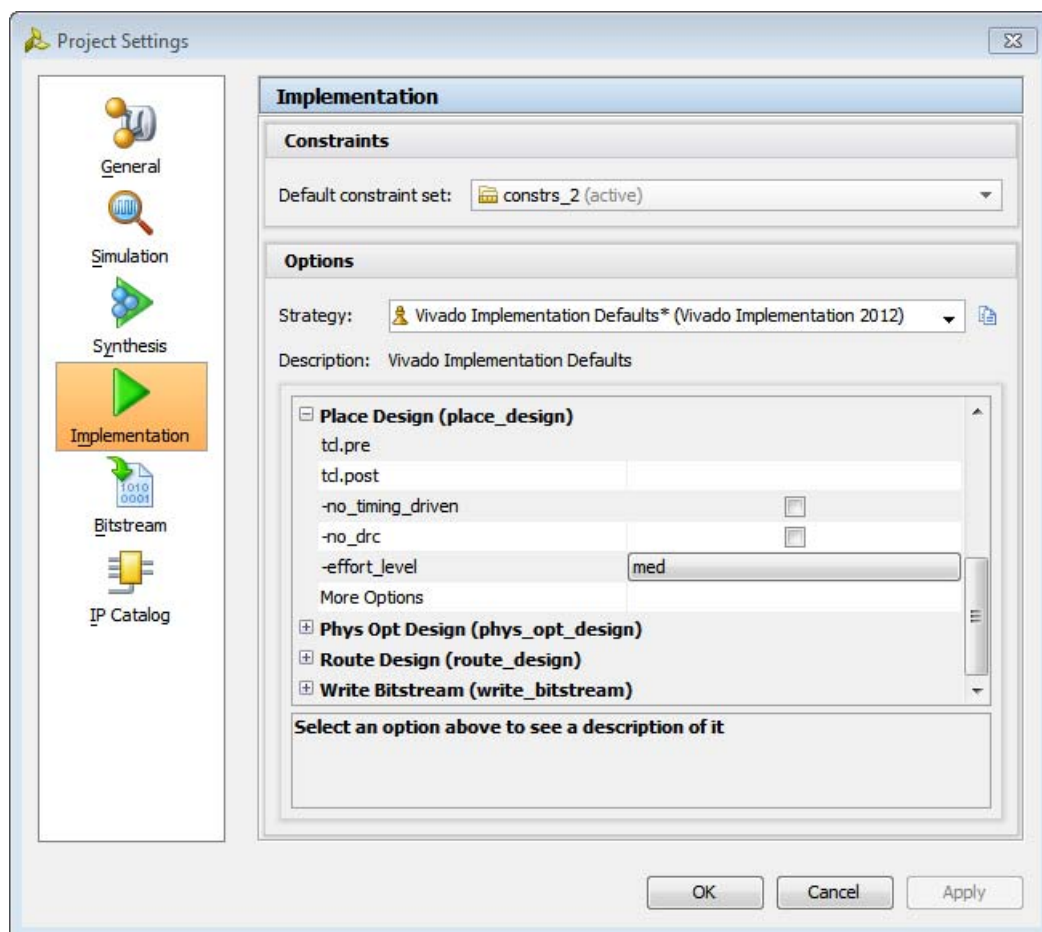


図 1-11 : [Project Settings] ダイアログ ボックスの [Implementation] ページ

## ストラテジの定義

ストラテジは、デザインの合成またはインプリメンテーションで最適な結果が得られるようにするために定義されたソリューションです。Vivado インプリメンテーション機能のあらかじめ設定されたオプションにより定義されます。ストラテジは、ツールおよびバージョン特定です。各メジャー リリースには、そのバージョン専用のストラテジがあります。

Vivado インプリメンテーションには、内部ベンチマークでテストされた一般的なストラテジが複数含まれています。これらの定義済みストラテジの設定は変更できませんが、提供されているストラテジをコピーし、それを変更することはできます。

現在定義されているストラテジを表示するには、メイン メニューから [Tools] → [Options] をクリックします。図 1-12 に、Vivado ツールで提供されるデフォルトのストラテジを示します。

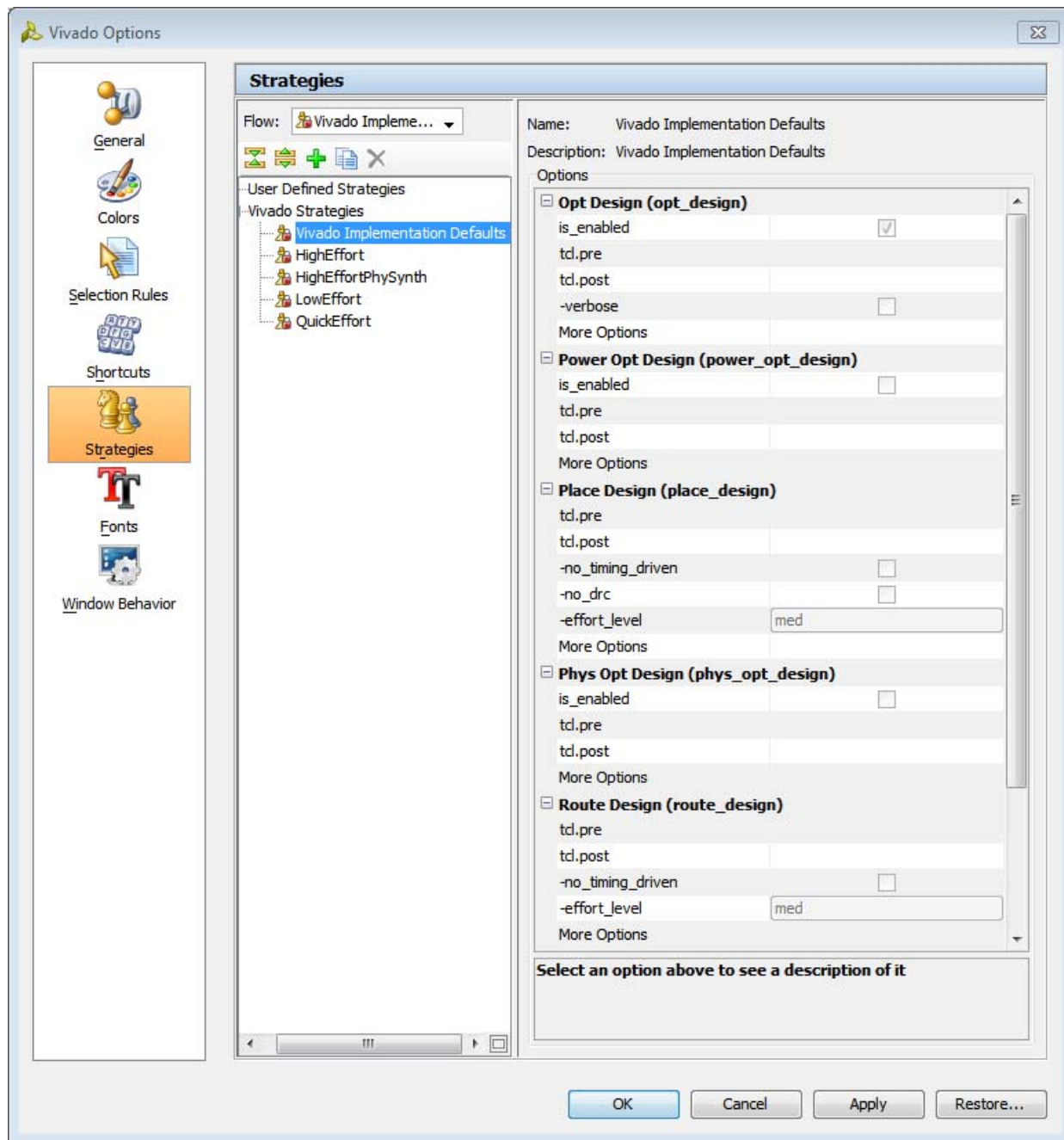


図 1-12 : デフォルトのインプリメンテーション ストラテジ

ストラテジを確認、コピー、変更するには、次の手順に従います。

1. [Tools] → [Options] をクリックします。
2. 左側のペインで [Strategies] をクリックします。図 1-12 に示す [Vivado Options] ダイアログ ボックスの [Strategies] ページに、各ツールとリリース バージョン用にあらかじめ定義されたストラテジがリストされます。
3. [Flow] ドロップダウン リストで、適切なバージョンの [Vivado Implementation] を選択します。提供されているストラテジが表示されます。



- 新しいストラテジを作成するには、ツールバーまたはポップアップ メニューから [Create New Strategy] をクリックします。



または、ツールバーまたはポップアップ メニューから [Create a Copy of this Strategy] をクリックして、既存のストラテジをコピーすることもできます。選択したストラテジが [User Defined Strategies] リストにコピーされ、右側にコマンド ライン オプションがリストされます。



- 新しいストラテジに対して次の情報を入力します。
  - [Name]: ストラテジの名前を入力します。
  - [Type]: [Synthesize] または [Implement] を指定します。
  - [Tool version]: ツール バージョンを指定します。
  - [Description]: ストラテジの説明を入力します。ここで入力した説明が [Design Run] ビューの結果の表に表示されます。
- インプリメンテーションの各段階 (opt\_design、power\_opt\_design、place\_design、phys\_opt\_design、route\_design、および write\_bitstream) のオプションを指定します。

The dialog box titled "New Strategy" contains the following fields:

- Name:** Strategy\_1
- Type:** Implement
- Tool version:** Vivado Implementation 2012
- Description:** (empty)

Buttons: OK, Cancel

特定のコマンド オプションをクリックすると、そのオプションの簡単な説明が下に表示されます。各インプリメンテーション段階の詳細と設定可能なオプションは、第 2 章「インプリメンテーション コマンド」を参照してください。

コマンド オプションを変更するには、そのコマンド オプションの右側をクリックします。

- 定義済みの値から選択するオプションは、ドロップダウン リストから選択します。
- イネーブル / ディスエーブルにするオプションは、チェック ボックスのオン/オフを切り替えます。
- テキスト入力フィールドのオプションは、値を入力します。
- ファイル名およびパスを指定するオプションは、ダイアログ ボックスでファイルを選択します。
- インプリメンテーションの各段階の最初 (tcl.pre) と最後 (tcl.post) に、カスタム Tcl スクリプトを挿入します。これにより、最適化前後にレポートを生成してタイミング結果を比較するなど、インプリメンテーション段階の前後に特定のタスクを実行できます。

The dialog box titled "Place Design (place\_design)" shows the following options:

- tcl.pre
- tcl.post
- no\_timing\_driven (checked)
- no\_drc (checked)
- effort\_level: med (dropdown menu with options: low, med, high, quick)
- More Options
- Phys Opt Design (phys\_opt\_d)
- is\_enabled\* (checked)
- tcl.pre

- [Apply] をクリックし、[OK] をクリックして新しいストラテジを保存します。

新しいストラテジは、[User Defined Strategies] の下にリストされます。ユーザー定義のストラテジは、次の場所に保存されます。

- Linux OS  
\$HOME/.Xilinx/Vivado/strategies
- Windows 7  
C:\Users\<username>\AppData\Roaming\Xilinx\Vivado\strategies
- Windows XP  
C:\Documents and Settings\username\Application Data\Xilinx\Vivado\strategies

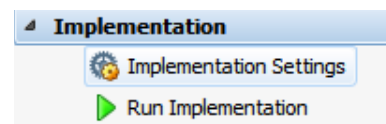


**ヒント:** 作成したストラテジを複数のユーザーで共有するには、ユーザー定義のストラテジを <InstallDir>/Vivado/<version>/strategies ディレクトリ (<InstallDir> はザイリンクス ツールのインストール ディレクトリ、<version> はリリース バージョン) にコピーします。

## run の実行

アクティブなインプリメンテーション run を実行するには、次のいずれかを実行します。

- Flow Navigator で [Run Implementation] をクリックします。
- メイン メニューから [Flow] → [Run Implementation] をクリックします。
- ツールバーの [Run Implementation] をクリックします。
- [Design Runs] ビューで run を右クリックし、ポップアップ メニューから [Launch Runs] をクリックします。



1 つのインプリメンテーション run を実行すると、そのインプリメンテーションに別のプロセスが生成されます。アクティブ run 以外の run を実行したり、[Design Runs] ビューで複数の run を選択して、複数のインプリメンテーション run を同時に実行したりもできます。

1. 複数の run を選択するには、Shift キーまたは Ctrl キーを押しながらクリックします。

[Design Runs] ビューで複数 run を選択する際、合成 run とインプリメンテーション run の両方を選択できます。Vivado IDE では、run の依存性が管理され、run が正しい順序で実行されます。

2. 右クリックして [Launch Runs] をクリックするか、[Design Runs] ビューのツールバーから [Launch Selected Runs] をクリックして、[Launch Selected Runs] ダイアログ ボックスを表示します (図 1-13)。

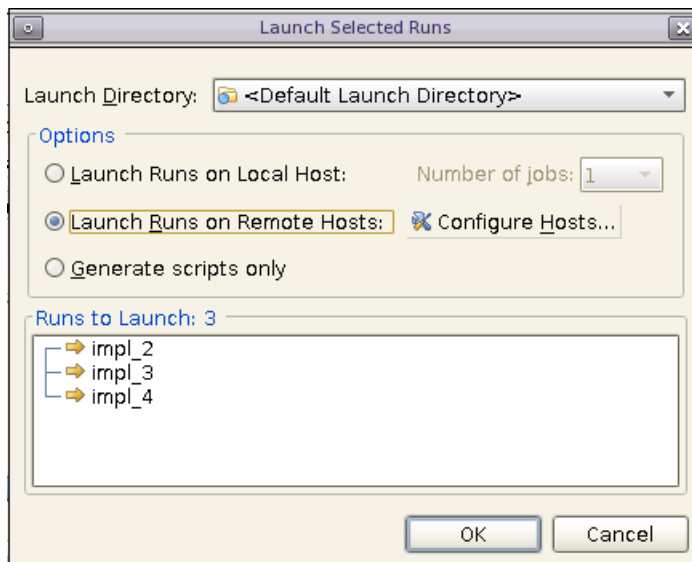


図 1-13 : [Launch Selected Runs] ダイアログ ボックス

- [Launch Directory]: デフォルトの起動ディレクトリは、ローカルプロジェクト ディレクトリ構造に含まれます。インプリメンテーション run のファイルは、次のディレクトリに保存されます。

`<project_name>/<project_name>.runs/<run_name>`



**ヒント:** プロジェクト ファイルには絶対パスが記述されるので、プロジェクト ディレクトリ外の場所を指定すると、プロジェクトを移動しにくくなります。



- 次のオプションを設定します。
  - [Launch Runs on Local Host] : run をローカル マシンで実行します。
    - [Number of jobs] : 複数の run を同時実行する際に使用するローカルプロセッサの数を指定します。各 run が各プロセッサで実行されます。Vivado では、インプリメンテーションのマルチスレッドはサポートされません。
  - [Launch Runs on Remote Hosts] (Linux のみ) : リモート ホストを使用してジョブを実行します。詳細は、付録 A 「リモート ホストの使用」を参照してください。
    - [Configure Hosts] : リモート ホストを設定します。
  - [Generate scripts only] : run ディレクトリおよび run スクリプトをエクスポートおよび作成しますが、run は実行しません。スクリプトは、Vivado IDE ツールの環境外で後で実行できます。

## プロセスのバックグラウンドへの移動

Vivado IDE で合成またはインプリメンテーションを実行するプロセスが生成されると、まず実行の準備としてデザイン ファイルと制約ファイルが読み込まれます。図 1-14 に示す [Starting Run] ダイアログ ボックスが表示され、このプロセスをバックグラウンドに移動し、CPU リソースを解放するオプションが示されます。

このプロセスをバックグラウンドに移動すると、バックグラウンド タスクを実行させたまま、レポートを表示したり、デザイン ファイルを開いたりといった別の操作を実行できます。run を実行しながら、前の run の結果を確認したり、レポートを表示したりして、時間を有効に活用できます。ただし、[Tcl Console] はブロックされるので、Tcl コマンドを使用したり、開いている別のデザインに切り替えるなどの Tcl コマンドを必要とするタスクは実行できません。

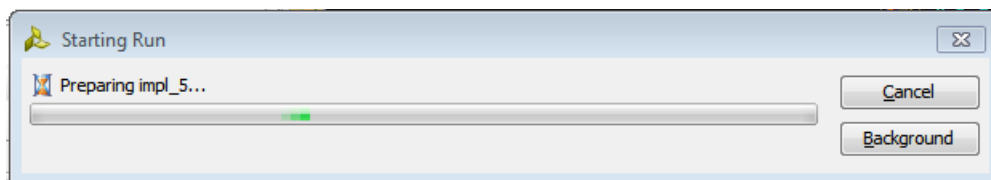


図 1-14 : [Starting Run] ダイアログ ボックス

## インプリメンテーションの段階ごとの実行

Vivado インプリメンテーションは、ロジック最適化、配置、配線などの複数のプロセスで構成されます。Vivado ツールでは、インプリメンテーションを 1 つのプロセスとしてではなく、1 つずつ順に実行することも可能です。

1. [Design Runs] ビューで 1 つの run を右クリックします。
2. ポップアップ メニューから [Launch Next Step: <Step>] をクリックします。<Step> の値は、次のとおりです。
  - opt\_design (デザインの最適化) : ザイリンクス FPGA にフィットするよう論理デザインを最適化します。
  - power\_opt\_design (デザインの消費電力最適化) : インプリメント済み FPGA の消費電力を削減するため、デザイン エレメントを最適化します。
  - place\_design (デザインの配置) : デザインをターゲット ザイリンクス デバイスに配置します。
  - phys\_opt\_design (デザインの物理最適化) : ファンアウトの大きいネットのドライバーを複製してロードを分散することにより、デザインのタイミングを最適化します。
  - route\_design (デザインの配線) : デザインをターゲット ザイリンクス デバイスに配線します。
  - write\_bitstream (ビットストリームの生成) : ザイリンクス デバイス コンフィギュレーションのビットストリームを生成します。ビットストリーム生成は、厳密にはインプリメンテーション run の一部ではありませんが、次の段階として実行可能です。

3. [Launch Next Step: <Step>] を繰り返し、インプリメンテーションの段階を進めていきます。

▶ Launch Next Step: phys\_opt\_design  
◀ Reset to Previous Step: place\_design

インプリメンテーション段階の間で必要に応じてレポート生成または解析を実行し、デザイン オプションの結果を確認できます。

4. 完了している直前の段階に戻すには、[Design Runs] ビューで run を右クリックして [Reset to Previous Step: <Step>] をクリックします。

このコマンドは、選択した run を現在のステートから直前の段階にリセットします。これにより、run を前の段階に戻して、必要に応じて変更を加えてから、run を次の段階へ進めて終了させることができます。

## インプリメンテーション run の監視

合成 run またはインプリメンテーション run のステータスは、[Log] ビュー (コンパイル情報)、[Messages] ビュー (情報、警告、エラー メッセージ)、[Project Summary] ビュー、または [Design Runs] ビューで確認できます。

### run ステータス表示の使用

進行中の run のステータスは、2 つの方法で表示されます。図 1-15 に示すこれらのステータス表示には、run が進行中であることが示されるほか、ここから必要に応じてキャンセルできるようになっています。

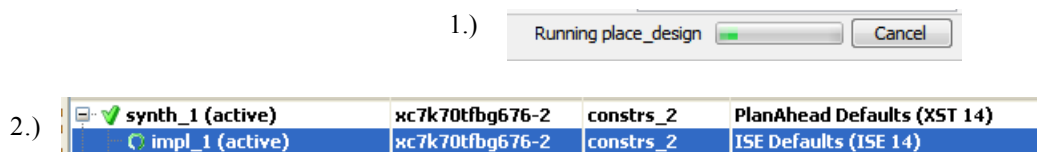


図 1-15: インプリメンテーション run のステータス表示

- Vivado IDE の右上にあるプロジェクト ステータス バーに表示される run ステータス インジケータでは、run が進行中であることが動くバーで示されます。[Cancel] ボタンをクリックすると、run を停止できます。
- 図 1-15 の下部に示す [Design Runs] ビューの run ステータス インジケータには、run が進行中であることを示す円形の矢印が表示されます。run を右クリックしてポップアップメニューから [Reset Run] をクリックすると、run をキャンセルできます。

### run のキャンセル/リセット

進行中の run を [Cancel] ボタンまたは [Reset Run] コマンドでキャンセルすると、キャンセルした run 用に作成された run ファイルをすべて削除するかどうか確認するメッセージが表示されます。

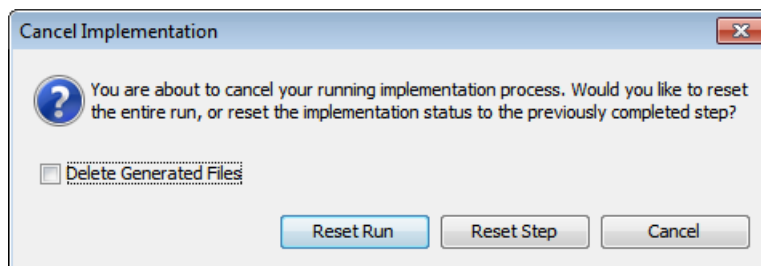


図 1-16: run のキャンセル

[Delete Generated Files] をオンにすると、ローカルプロジェクト ディレクトリから run データが削除されます。キャンセルした run で作成されたデータはすべて削除して、今後の run で問題にならないようにしておくことをお勧めします。

## ログの表示

run を実行すると、[Log] ビューが開き、標準出力メッセージが表示されます。図 1-17 に、[Log] ビューの例を示します。

[Log] ビューには、place\_design や route\_design などの各インプリメンテーションプロセスの進行状況が表示されます。異なるメッセージがどこから表示されているかを確認し、インプリメンテーション run のデバッグに役立てることができます。

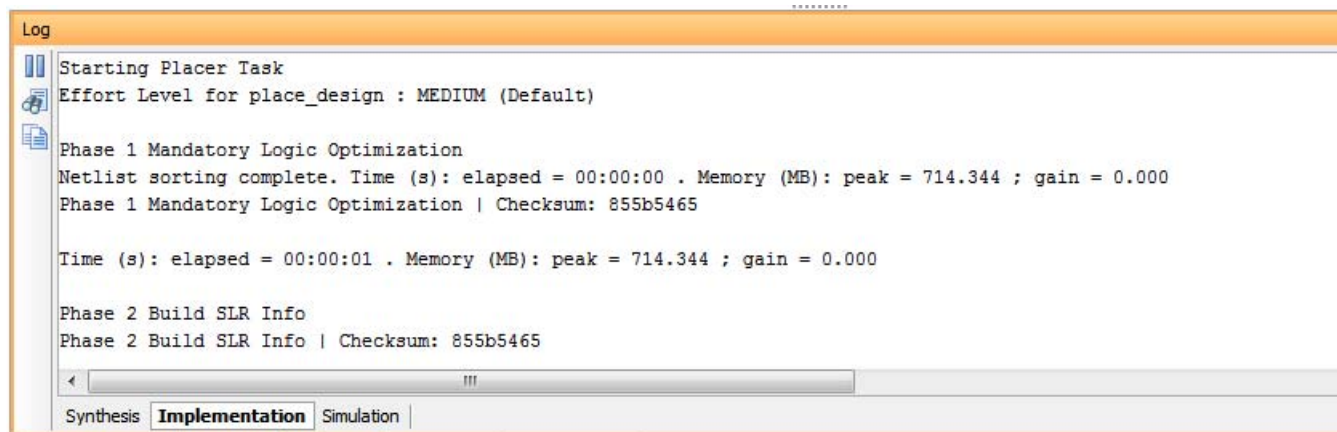


図 1-17: [Log] ビュー

[Pause output] ボタンをクリックすると、[Log] ビューの出力を一時停止でき、コマンドの実行中にログをスクロールして読むことができます。

## プロジェクト ステータスの確認

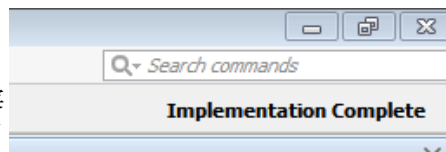
Vivado IDE では、複数の方法で、プロジェクトの全体的なステータスと、プロセスの次の手順を実行する方法が示されます。プロジェクト ステータスには、デザインプロセスの主なタスクの結果のみが示されます。

プロジェクトの全体的なステータスは [Project Summary] ビューとステータス バーに表示され、プロジェクトを開いたとき、デザインフロー コマンドを実行したときにプロジェクトのステータスをすばやく判断できます。RTL エラボーレーション、合成、インプリメンテーション、ビットストリーム生成などのステータスが示されます。

### プロジェクト ステータス バー

プロジェクトの全体的なステータスは、Vivado IDE の右上のプロジェクト ステータス バーに表示されます。

エラボーレーション、合成、インプリメント、ビットストリームの生成を実行すると、プロジェクト ステータス バーにその結果が示されます。プロセスでエラーが発生した場合は、赤色の文字で表示されます。



合成またはインプリメンテーションを完了した状態でソース ファイルまたはデザイン制約を変更すると、プロジェクト ステータスが図 1-18 に示すように「Out-of-Date」と表示されます。これは、プロジェクトが最新でなく、更新が必要であることを示します。デザインのどの部分が最新ではないのかを確認するには、[more info] リンクをクリックします。インプリメンテーションのみの再実行が必要な場合や、合成およびインプリメンテーションの両方を再実行する必要がある場合があります。

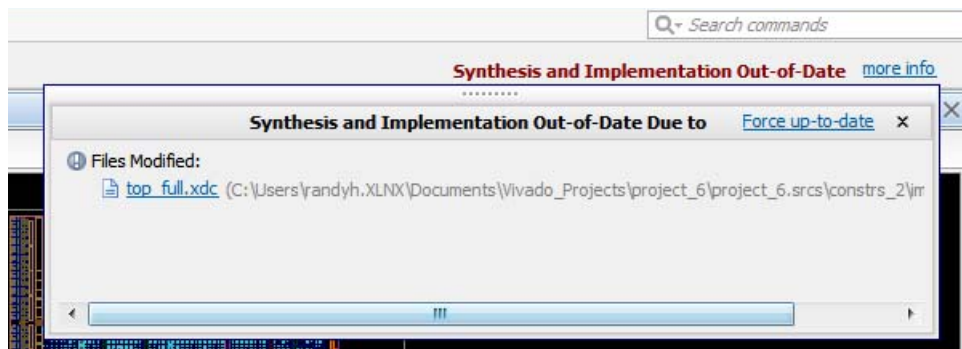


図 1-18: インプリメンテーションが最新でない

図 1-18 に示す [more info] リンクをクリックしたときに表示されるダイアログ ボックスには、[Force-up-to-date] リンクがあり、インプリメンテーション run または合成 run のステータスを強制的に最新の状態にすることができます。この機能は、デザインまたは制約を変更したが、現在の run の結果を解析する場合などに使用します。[Force-up-to-date] コマンドは、[Design Runs] ビューで最新でない run を右クリックしたときに表示されるポップアップ メニューにもあります。詳細は、『Vivado Design Suite ユーザー ガイド : Vivado IDE の使用』(UG893) を参照してください。

## インプリメンテーション完了後の次の操作

プロジェクト モードおよび非プロジェクト モードのどちらでも、インプリメンテーションが完了した後の操作は、インプリメンテーションの結果によります。デザインが完全に配置配線されたか、解決が必要な問題があるのか、タイミング制約およびデザイン要件が満たされたか、デザインを完了するのに変更が必要か、ザイリンクス パーツ用のビットストリームを生成してもよいかなどです。

非プロジェクト ベースのデザインでは、デザイン セッションで生成されたメッセージは Vivado ログ ファイル (vivado.log) に保存されます。このログ ファイルおよびデザイン データからのレポートを参照し、プロジェクトの状況を正確に評価します。

プロジェクト ベースのデザインでは、ログ ファイルのメッセージが [Messages] ビューに表示され、確認する必要がある多くのレポートが自動的に生成されます。プロジェクト モードでは、インプリメンテーション run が完了すると、図 1-19 に示すダイアログ ボックスが表示され、次の操作を選択できます。

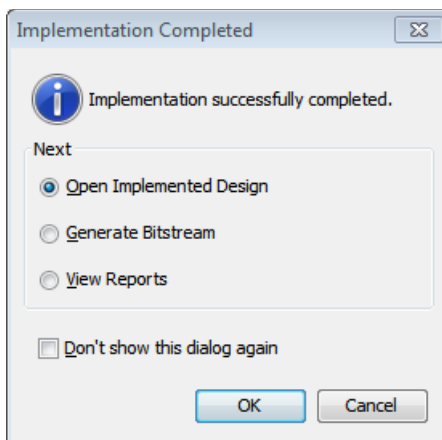


図 1-19: プロジェクト モード : [Implementation Completed] ダイアログ ボックス

[Implementation Completed] ダイアログ ボックスで次のいずれかのオプションをオンにし、[OK] をクリックします。

- [Open Implemented Design]: ネットリスト、デザイン制約、ターゲット パーツ、配置配線の結果を Vivado IDE に読み込み、必要に応じてデザインを解析できるようにします。
- [Generate Bitstream]: [Generate Bitstream] ダイアログ ボックスを開きます。詳細は、『Vivado Design Suite ユーザーガイド: プログラムおよびデバッグ』(UG908) を参照してください。
- [View Reports]: Vivado ツールでインプリメンテーション中に生成されたレポート ファイルを選択して表示できる [Reports] ビューを開きます。詳細は、30 ページの「インプリメンテーション レポートの表示」を参照してください。

インプリメンテーション後は、次の手順を実行することをお勧めします。

1. インプリメンテーション メッセージを確認します。
2. インプリメンテーション レポートを表示し、次の事項を確認します。
  - 。 タイミング制約が満たされている (report\_timing\_summary)。
  - 。 リソース使用率が予測どおりである (report\_utilization)。
  - 。 消費電力が予測どおりである (report\_power)。
3. ビットストリーム ファイルを生成します。これには、デザインがハードウェアのルールに違反していないことを確認する最終 DRC も含まれます。

満たされていないデザイン要件がある場合は、プロジェクト モードではインプリメント済みデザインを開いて解析できます。非プロジェクト モードでは、インプリメンテーション後のデザイン チェックポイントを開きます。インプリメント済みデザインの解析については、『Vivado Design Suite ユーザー ガイド: デザイン解析およびクロージャ テクニック』(UG906) を参照してください。

## メッセージの表示



**重要:** すべてのメッセージを確認してください。メッセージには、デザインのパフォーマンス、消費電力、エリア、配線を向上するための推奨事項が記載される場合があります。クリティカル警告メッセージにも、解決すべきタイミング制約の問題が表示されることがあります。

非プロジェクト モードでは、Vivado ツール ログ ファイル (vivado.log) でメッセージを確認します。このファイルには、1 回のデザイン セッションで使用したすべてのコマンドおよびコマンドの結果と生成されたメッセージが保存されます。Vivado テキスト エディターでログ ファイルを開き、すべてのコマンドの結果を確認してください。有益な情報が得られることがあります。

プロジェクト モードでは、[Messages] ビューに [Log] ビューの内容がフィルターされたものが表示され、主なメッセージ、警告、およびエラー メッセージのみが含まれます。[Messages] ビューは機能ごとに分類されており、フィルターを適用するツールバー オプションを使用して、特定のタイプのメッセージのみを表示できます。図 1-20 に、[Messages] ビューの例を示します。

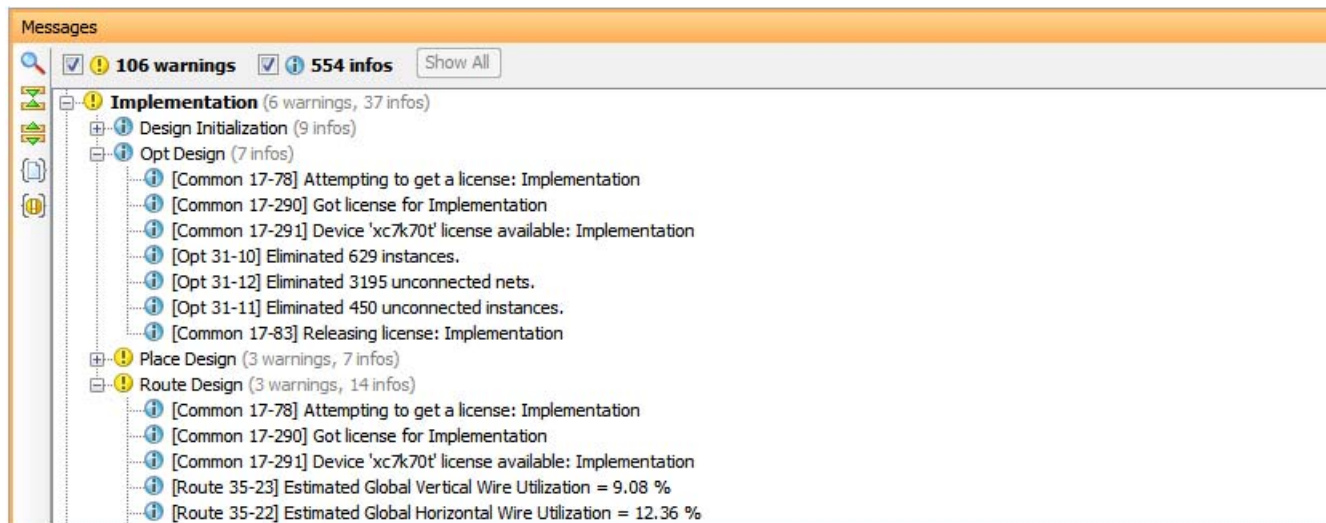


図 1-20 : [Messages] ビュー

横にあるプラス記号 (+) をクリックして展開し、各メッセージを表示します。図 1-20 に示す [Messages] ビューのパナーで、エラー、クリティカルな警告、警告、情報メッセージのチェックボックスをオンにすると、該当するメッセージが表示されます。

[Messages] ビューでリンクを含むメッセージを選択すると、ソースファイルが開き、該当する行がハイライトされます。メッセージを右クリックして [Search for Answer Record] をクリックすると、ザイリンクス ウェブサイトでそのメッセージに関連するアンサー データベースを検索できます。

## インプリメンテーション レポートの表示

Vivado ツールでは、デザイン データからさまざまなレポートを生成できます。タイミング、タイミング コンフィギュレーション、タイミング サマリ、クロック、クロック ネットワーク、クロック使用率、消費電力、スイッチング アクティビティ、ノイズ解析などのレポートを生成できます。これらは、Vivado ツールで生成できるレポートのごく一部です。

レポートを表示すると、次を実行できます。

- スクロール バーを使用してレポート ファイルを参照
- [Show Find] または [Find in Files] ボタンをクリックし、特定テキストを検索
- [Move Caret to Document Start] または [Move Caret to Document End] ボタンをクリックしてファイルの冒頭または最後に移動



## 非プロジェクト モードでのレポート生成

非プロジェクト モードでは、Tcl コマンドを使用してレポートを手動で生成する必要があります。Tcl スクリプトで複数のレポートを生成することもできます。次のスクリプト例では、複数のレポートを生成し、Reports フォルダに保存しています。

```
# Report the 15 longest carry chains
report_carry_chains -file C:/Reports/carry_chains.rpt -max_chains 15
# Run Timing Summary Report for post implementation timing
report_timing_summary -file C:/Reports/post_route_timing.rpt -name time1
# Run Utilization Report for device resource utilization
report_utilization -file C:/Reports/post_route_utilization.rpt
```

これらのレポートを Vivado IDE で開くこともできます。上記のスクリプト例では、report\_timing\_summary コマンドに -file オプションと -name オプションの両方を使用し、レポートをファイルおよび IDE のビューに表示しています。32 ページの図 1-22 に、Vivado IDE でレポートを開いた例を示します。



**ヒント：**レポートを保存するディレクトリは、レポートの生成を実行する前に存在している必要があります。ディレクトリがないと、ファイルは保存されません。

Tcl レポート コマンドとそのオプションについては、Vivado IDE または Tcl コマンド プロンプトで Tcl コマンド help を使用するか、『Vivado Design Suite Tcl コマンド リファレンス ガイド』(UG835) を参照してください。

## プロジェクト モードでのレポート生成

プロジェクト モードでは、デザイン フローを実行していくと多くのレポートが自動的に生成されます。Vivado ツールで生成されたレポート ファイルは、Vivado IDE の [Reports] ビュー (図 1-21) から表示できます。このビューは、合成またはインプリメント コマンドを実行すると自動的に開きます。このビューが開いていない場合は、[Project Summary] ビューで [Reports] リンクをクリックするか、[Windows] → [Reports] をクリックします。



**ヒント：**インプリメンテーション run の tcl.pre おおび tcl.post オプションを使用すると、プロセスの各段階でカスタム レポートを出力できます。これらのカスタム レポートは [Reports] ビューには表示されませんが、ニーズに合わせてカスタマイズできます。詳細は、17 ページの「インプリメンテーション run 設定の変更」を参照してください。

Name	Modified	Size
<b>Synth Design (synth_design)</b>		
Vivado Synthesis Report	7/10/12 9:29 AM	423.3 KB
Utilization Report	7/10/12 9:29 AM	7.0 KB
<b>Place Design (place_design)</b>		
Place and Route Log	7/11/12 6:50 PM	21.7 KB
IO Report	7/11/12 6:44 PM	146.4 KB
Clock Utilization Report	7/11/12 6:44 PM	20.9 KB
Utilization Report	7/11/12 6:44 PM	9.3 KB
Control Sets Report	7/11/12 6:44 PM	138.5 KB
<b>Route Design (route_design)</b>		
WebTalk Report		
DRC Report	7/11/12 6:49 PM	29.9 KB
Power Report	7/11/12 6:50 PM	30.9 KB
Route Status Report	7/11/12 6:50 PM	0.6 KB
Timing Summary Report	7/11/12 6:50 PM	470.5 KB
<b>Write Bitstream (write_bitstream)</b>		
WebTalk Report		

図 1-21 : [Reports] ビュー

[Reports] ビューから表示されるレポートはテキスト形式で、run に関する情報が含まれます。[Reports] ビューでレポートをダブルクリックすると、別のビューにテキスト形式のレポートが表示されます (図 1-22)。

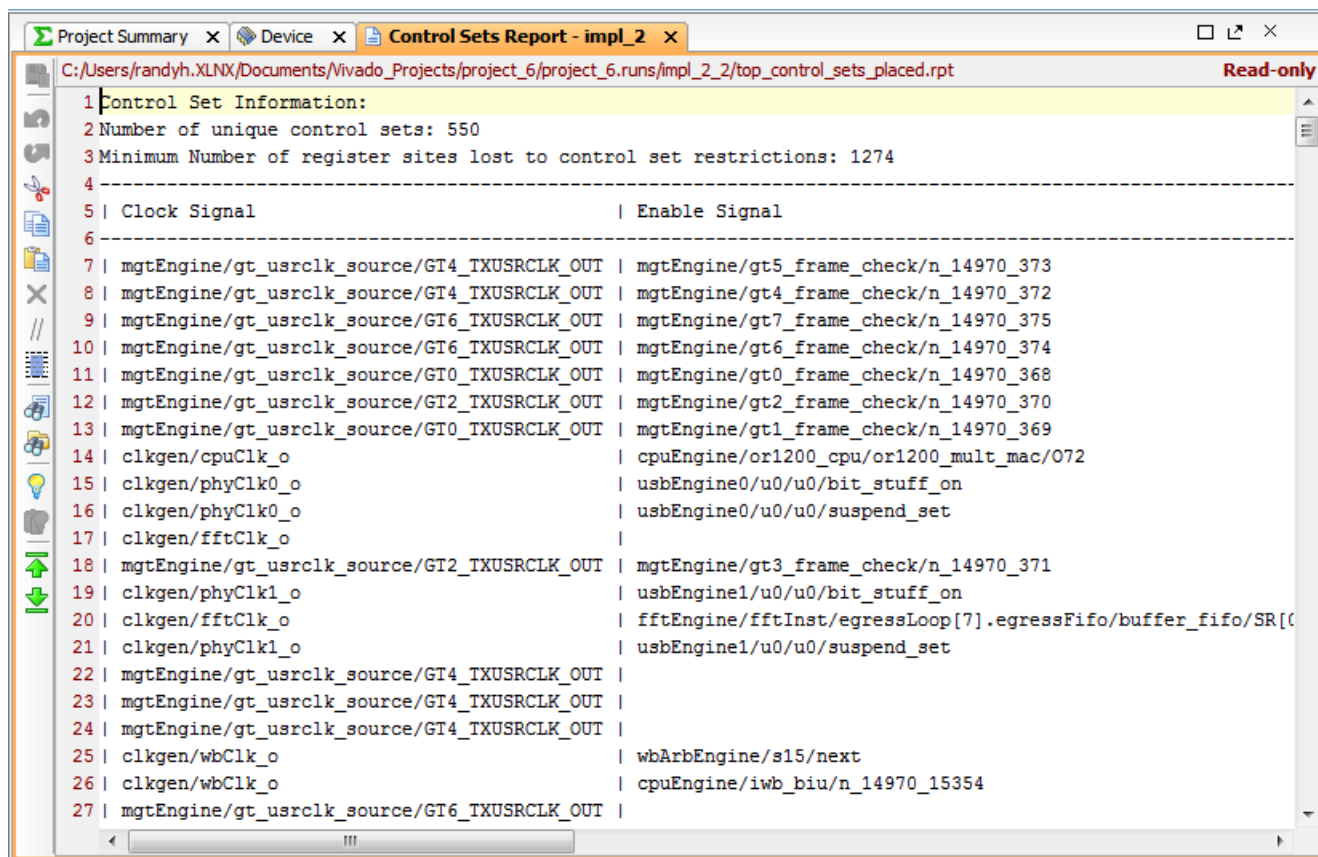


図 1-22: 制御セット レポート

## レポートからのクロスプローブ

プロジェクト モードおよび非プロジェクト モードの両方で、レポートと [Device] ビューなどのビューに表示される関連デザイン データのクロスプローブがサポートされています。ただし、この機能はテキスト形式のレポートでは使用できません。メニュー コマンドまたは Tcl コマンドを使用してレポートを生成する必要があります。



**ヒント:** Vivado IDE では、レポートとデザイン ビューのクロスプローブがプロジェクト モードおよび非プロジェクト モードの両方でサポートされています。

たとえば、図 1-21 に示すように、[Reports] ビューには [Route Design] の下にテキスト形式のタイミング サマリ レポートが含まれています。タイミングを解析する際は、クリティカル パスの配置および配線リソースなどのデザイン データを [Device] ビューに表示すると有益です。Vivado IDE で [Tools] → [Timing] → [Report Timing Summary] をクリックしてレポートを生成すると、表示されるレポートからデザインの異なるビューにクロスプローブできます。

33 ページの図 1-23 に、タイミング サマリ レポートと [Device] ビューのクロスプローブの例を示します。この非プロジェクト モードの例では、配線後のデザイン チェックポイントを Vivado IDE で開き、デザイン サマリ レポートを生成して IDE で開いています (report\_timing\_summary -name コマンドを使用)。[Device] ビューでは [Routing



Resources] がオンになっています。タイミング サマリ レポートでタイミング パスを選択すると、[Device] ビューでも選択されます。

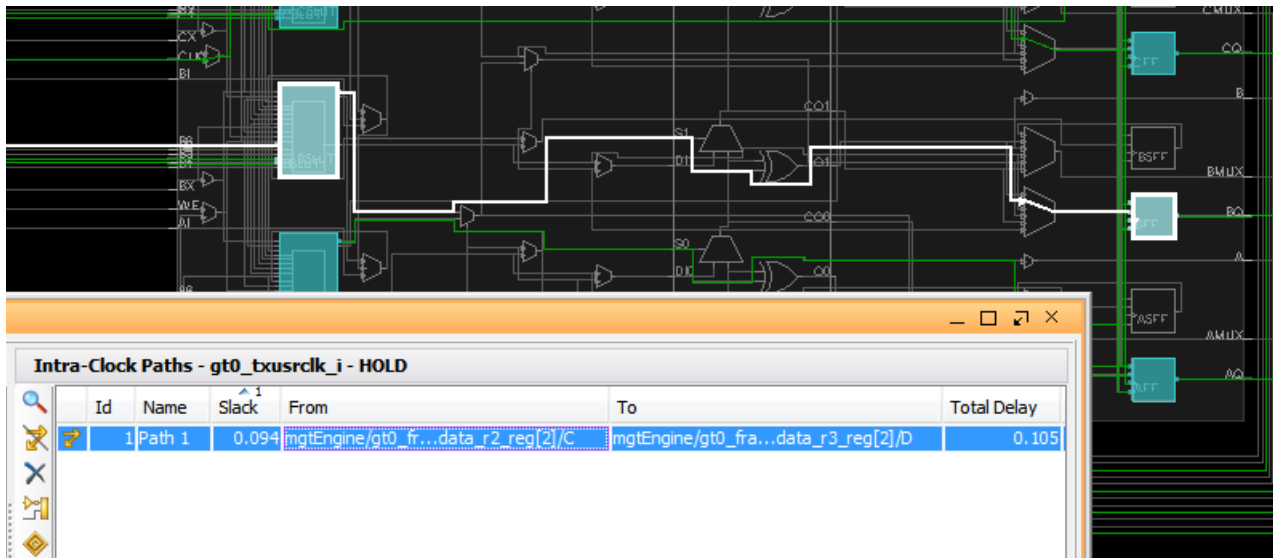


図 1-23 : タイミング レポートと [Device] ビューのクロスプローブ

レポート解析およびデザイン クロージャのストラテジについては、『Vivado Design Suite ユーザー ガイド : デザイン 解析およびクロージャ テクニック』(UG906) を参照してください。

# インプリメンテーション コマンド

---

## 概要

Vivado IDE では、13 ページの「プロジェクト モードでのインプリメンテーションの実行」に説明するようにインプリメンテーション プロセスを手動で段階ごとに実行する機能など、プロジェクト ベースのデザインでインプリメンテーション プロセスを管理および簡略化する多数の機能が提供されています。非プロジェクト モードでは、11 ページの「非プロジェクト モードでのインプリメンテーションの実行」に説明するように、Tcl コマンドを使用してインプリメンテーション プロセスの各段階を手動で実行する必要があります。

プロジェクト モードおよび非プロジェクト モードのどちらでも、Vivado インプリメンテーション プロセスはいくつかのサブプロセスで構成されています。

- 合成済みデザインを開く：ネットリスト、デザイン制約、ターゲット デバイス データを統合し、インプリメンテーションを実行するデザインをメモリ内に構築します。
- `opt_design` (デザインの最適化)：ザイリンクス FPGA にフィットするよう論理デザインを最適化します。
- `power_opt_design` (デザインの消費電力最適化)：インプリメント済み FPGA の消費電力を削減するため、デザインエレメントを最適化します。この手順はオプションです。
- `place_design` (デザインの配置)：デザインをターゲット ザイリンクス デバイスに配置します。
- `phys_opt_design` (デザインの物理最適化)：ファンアウトの大きいネットのドライバーを複製してロードを分散することにより、デザインのタイミングを最適化します。この手順はオプションです。
- `route_design` (デザインの配線)：デザインをターゲット ザイリンクス デバイスに配線します。
- `write_bitstream` (ビットストリームの生成)：ザイリンクス デバイス コンフィギュレーションのビットストリームを生成します。ビットストリーム生成は、厳密にはインプリメンテーション `run` の一部ではありませんが、別の段階として実行可能です。

この章では、インプリメンテーション プロセスの各段階と関連する Tcl コマンドの詳細を説明します。すべての Tcl コマンドおよびそのオプションの詳細は、『Vivado Design Suite Tcl コマンド リファレンス ガイド』(UG835) を参照してください。

## 合成済みデザインを開く

インプリメンテーションでは、まず最初に合成済みデザインからのネットリストをメモリに読み込み、デザイン制約を適用します。合成済みのデザインは、使用されるデザイン フローによってさまざまな方法で開くことができます。プロジェクト モードおよび火プロジェクト モードの両方で、デザインのソース ファイルおよびステートに応じて、次の Tcl コマンドを使用して合成済みデザインをメモリに読み込むことができます。

- `synth_design`
- `read_checkpoint`
- `open_run`
- `link_design`

メモリ内にデザインを構築するため、次のプロセスでネットリスト ファイル、制約ファイル、およびターゲット デバイスの情報が統合されます。

1. 必要であれば、複数のソースからのネットリストを統合します。デザインには、構造 Verilog、EDIF、NGC を含めることができます。
2. 従来のネットリスト プリミティブを現在サポートされる Unisim プリミティブに変換します。



**ヒント** : `report_transformed_primitives` を使用すると、変換されたセルのリストを生成できます。

3. セルの形を作成します。Vivado ツールでは、接続または配置制約に基づいてセルの非明示的な形が作成され、配置が簡略化されます。相対配置マクロ (RPM) は、非明示的な形の例です。RPM は、個別のセルではなくグループとして配置されます。非明示的な形の例には、複数のスライスに配置する必要のある長いキャリー チェーンもあります。キャリー チェーンを構成する CARRY4 エLEMENT は 1 つの形に含め、縦方向に並ぶ複数のスライスに配置されるようにする必要があります。

### synth\_design

このコマンドは、RTL ソースに対して指定したオプションで Vivado 合成を実行し、合成後にデザインをメモリに読み込みます。synth\_design コマンドは、プロジェクト モードおよび非プロジェクト モード両方で使用できます。

```
synth_design [-name <arg>] [-part <arg>] [-constrset <arg>] [-top <arg>]
  [-flatten_hierarchy <arg>] [-gated_clock_conversion <arg>] [-effort_level <arg>]
  [-rtl] [-no_iobuf] [-bufg <arg>] [-fanout_limit <arg>] [-fsm_extraction <arg>]
  [-quiet] [-verbose]
```

#### サンプルスクリプト

次のコードは、Vivado ツールのインストールディレクトリの `examples/Vivado_Tutorials` ディレクトリに含まれる `create_bft_batch.tcl` スクリプトからの抜粋です。スクリプトの使用方法は、『Vivado Design Suite ユーザーガイド : デザイン フローの概要』(UG892) を参照してください。

```
# Setup design sources and constraints
read_vhdl -library bftLib [ glob ./Sources/hdl/bftLib/*.vhd ]
read_vhdl ./Sources/hdl/bft.vhdl
read_verilog [ glob ./Sources/hdl/*.v ]
read_xdc ./Sources/bft_full.xdc

# Run synthesis, report utilization and timing estimates, write design checkpoint
synth_design -top bft -part xc7k70tfbg484-2 -flatten rebuilt
write_checkpoint -force $outputDir/post_synth
```

このサンプル スクリプトでは、VHDL および Verilog ファイルが読み込まれ、指定したデバイスでデザインが合成されます。synth\_design コマンドが完了すると、デザインがメモリに読み込まれます。合成が完了すると、デザイン チェックポイントが保存されます。

すべての Tcl コマンドおよびそのオプションの詳細は、『Vivado Design Suite Tcl コマンド リファレンス ガイド』(UG835)を参照してください。

## read\_checkpoint

非プロジェクト モードで、合成済みデザイン チェックポイントを開きます。

```
read_checkpoint [-part <arg>] [-quiet] [-verbose] <file>
```

### サンプル スクリプト

```
# Read the specified design checkpoint and create an in-memory design.
read_checkpoint C:/Data/post_synth.dcp
```

このサンプル スクリプトでは、合成済みデザイン チェックポイントが開きます。

## open\_run

合成済みまたはインプリメント済み run を開き、メモリに読み込みます。デザイン run は非プロジェクト モードではサポートされていないので、このコマンドはプロジェクト モードでのみ機能します。

このコマンドをインプリメンテーション前の RTL デザインに使用し、完了した Vivado 合成または XST run を開いて、合成済みネットリストをメモリに読み込みます。synth\_design を実行した後は、メモリ内のデザインが自動的にアップデートされるので、このコマンドを実行する必要はありません。このコマンドは、以前のデザイン セッションで完了した合成 run を開く場合にのみ使用します。

**注記:** このコマンドは、RTL デザイン用です。ネットリスト ベースのデザインを開くには、link\_design コマンドを使用します。

```
open_run [-name <arg>] [-quiet] [-verbose] <run>
```

### サンプル スクリプト

```
# Open named design from completed synthesis run
open_run -name synth_1 synth_1
```

このサンプル プロジェクトでは、synth\_1 というデザインを開き、synth\_1 という完了した合成 run をメモリに読み込みます。

## link\_design

サードパーティ合成ツールなどで生成されたネットリスト ソースからメモリにデザインを作成し、ネットリストと合成制約をターゲット デバイスにリンクします。このコマンドは、プロジェクト モードおよび非プロジェクト モードの両方でネットリスト デザインを作成する場合にサポートされます。

```
link_design [-name <arg>] [-part <arg>] [-constrset <arg>] [-top <arg>]
            [-quiet] [-verbose]
```

### サンプル スクリプト

```
# Open named design from netlist sources.
link_design -name netDriven -constrset constrs_1 -part xc7k325tfbg900-1
```

**注記:** デザインがメモリにある場合に open\_run または link\_design コマンドを実行すると、新しいデザインを開く前に、現在のデザインへの変更を保存するかどうかを尋ねるメッセージが表示されます。

メモリに合成済みデザインを作成した後、エラーおよびクリティカル警告を調べて、不足している制約や不正な制約がないかどうかを確認してください。デザインが問題なく作成されたら、解析、レポート生成、制約の適用、またはインプリメンテーションを実行できます。



**推奨:** メモリに合成済みデザインを開いたら、`report_timing_summary` コマンドを実行してタイミング制約をチェックし、デザイン目標が適切であるかどうかを確認すると有益です。

## ロジック最適化

ロジック最適化は、配置の前に効率的なロジック デザインを得るために実行します。ロジック最適化では、ネットリスト接続性チェックが実行され、複数のドライバーを持つネットや駆動されていない入力などの潜在的なデザイン問題に対して警告メッセージが表示されます。

Vivado ツールでは、メモリ内のデザインに対して次のロジック最適化がデフォルトで実行されます。

1. リターゲット: 最適化しやすくなるように、セル タイプを別のセル タイプに置換します。たとえば、MUXF7 はほかの LUT と統合できるように LUT3 に置換されます。また、インバーターのような単純なセルはダウンストリーム ロジックに吸収されます。
2. 定数伝搬: 定数値をロジックに伝搬すると、ロジックは次のようになります。
  - 。 削除 (例: 0 が入力される AND)
  - 。 縮小 (例: 1 が入力される 3 入力 OR を 2 入力 OR に削減)
  - 。 冗長 (例: 0 が入力される 2 入力 OR を 1 本のワイヤに削減)
3. リマップ: 複数の LUT を 1 つの LUT にまとめてロジックの深さを削減します。
4. スイープ: ロードのないセルを削除します。
5. エリア モードでの再合成: エリア モードで合成を再実行し、LUT の数を削減します。

### opt\_design



**重要:** 各ロジック最適化はメモリ内のデザインに適用され、元の合成済みデザインには適用されません。

現在のネットリストを最適化し、すべての最適化をデフォルトで実行します。opt\_design を実行すると、メモリ内のデザインが読み込まれ、最適化され、最適化されたデザインがメモリに戻されます。

```
opt_design [-sweep] [-retarget] [-propconst] [-remap] [-resynth <arg>]  
           [-mode <arg>] [-effort_level <arg>] [-quiet] [-verbose]
```

#### サンプルスクリプト

```
opt_design -retarget -propconst -sweep
```

コマンド ライン オプションを使用すると、最適化タイプを制限して実行できます。たとえば、密集したデザインにはリマップを実行しない方がよい場合があります。



**推奨:** 最適化結果を解析するには、`-verbose` オプションを使用し、opt\_design の最適化の影響を受けたロジックの詳細を確認します。`-verbose` オプションを使用すると大量のメッセージが表示されるので、デフォルトではオフになっています。

-verbose オプションを使用して `opt_design` コマンドを再実行すると最適化結果が変わるので、有益だと思われる場合は -verbose オプションを使用します。

```
opt_design
opt_design -verbose
```

## ロジック最適化制約

ロジック最適化中、`DONT_TOUCH` および `MARK_DEBUG` プロパティが認識され、これらのプロパティが設定されているネットは削除されません。詳細は、『Vivado Design Suite ユーザー ガイド：合成』(UG901)を参照してください。

- `MARK_DEBUG` : ChipScope™ Pro ツールを使用してプローブする予定のネットに設定します。`MARK_DEBUG` の設定されたネットはスライス境界に接続され、プローブできる状態になります。
- `DONT_TOUCH` : 下位セルに設定し、最適化されないようにします。階層セルに `DONT_TOUCH` を設定するとセルの境界は保持されますが、セル内で最適化が実行される可能性があります。



**重要 :** ISE Design Suite からデザインを移行した場合、`KEEP` および `KEEP_HIERARCHY` 制約が自動的に `DONT_TOUCH` 制約に変換されます。

## 消費電力の最適化

クロック ゲーティングを使用して、ダイナミック消費電力を最適化します (オプション)。消費電力の最適化は、プロジェクト モードおよび非プロジェクト モード両方で使用できます。詳細は、『Vivado Design Suite ユーザー ガイド：消費電力解析および最適化』(UG907)を参照してください。

Vivado の消費電力最適化は、ロジック最適化後に実行し、デザインの消費電力を最適化できます。消費電力の最適化には、クロックやロジックを変更せずに FPGA デザインのダイナミック消費電力を削減するクロック ゲーティングソリューションが含まれます。

レガシ IP ブロックおよびサードパーティ IP ブロックも含めたデザイン全体で解析が実行され、各クロック サイクルでの結果に影響しないレジスタからの出力ロジックが特定されます。7 シリーズ FPGA のロジックに多数含まれるクロック イネーブル (CE) が利用され、細粒度クロック ゲーティングまたはロジック ゲーティング信号が作成され、余分なスイッチング アクティビティが除去されます。また、フリップフロップ レベルでは、CE はフリップフロップの D 入力とフィードバック Q 出力のいずれかを選択するのではなく、クロックをゲーティングしているので、CE 入力のパフォーマンスが向上するだけでなく、クロックの消費電力も削減されます。

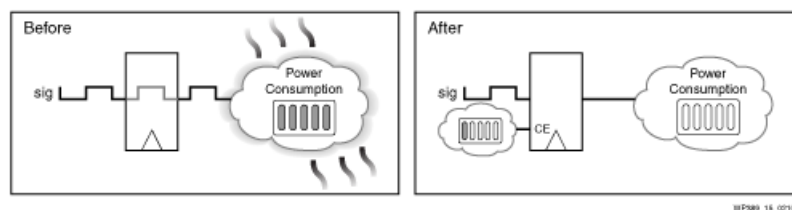


図 2-1: クロック ゲーティング

クロック ゲーティングでは、シングルデュアルポートまたは真のデュアルポート モード両方の専用ブロック RAM の消費電力も削減されます (39 ページの図 2-2)。これらのブロックには、アレイ イネーブル、ライト イネーブル、および出力レジスタのクロック イネーブルなどのイネーブル信号があります。節約される消費電力のほとんどはアレイ イネーブルの使用によるもので、データが書き込まれず、出力が使用されないときに、消費電力を削減する機能がインプリメントされます。

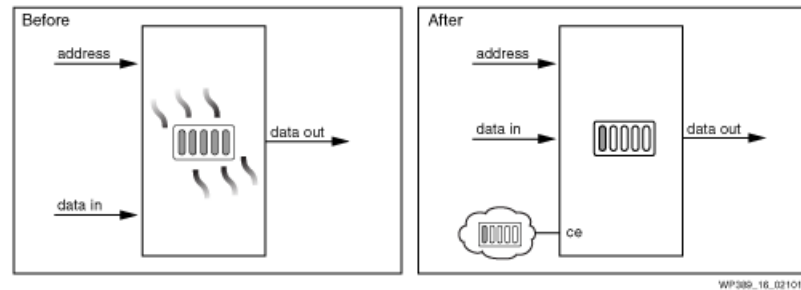


図 2-2 : BRAM イネーブルの利用

## power\_opt\_design

消費電力最適化を実行する前に、`set_power_opt` コマンドを使用して消費電力最適化コマンドを設定できます。これにより、最適化にセル タイプまたは階層を含めるか、除外するかなどを設定できます。このコマンドの構文は、次のとおりです。

```
set_power_opt [-include_cells <args>] [-exclude_cells <args>] [-clocks <args>]
               [-cell_types <args>] [-quiet] [-verbose]
```

`power_opt_design` コマンドでは、デザインが解析され、クロック ゲーティングにより消費電力が最適化されます。次の構文を使用します。

```
power_opt_design [-quiet] [-verbose]
```

### サンプルスクリプト

```
power_opt_design
```

`set_power_opt` コマンドで範囲を変更していない場合、デフォルトでデザイン全体が解析され、最適化されます。

## 配置

ロジック最適化および消費電力最適化 (オプション) が終了したら、次は配置を実行します。Vivado 配置エンジンによりネットリストのセルがターゲット デバイスの特定サイトに配置されます。ロジック最適化と同様、メモリ内のデザインに対して処理が実行されます。

Vivado の配置エンジンでは、次を最適化するよう配置が実行されます。

- タイミング スラック : タイミング クリティカルなパスのセルの配置は、負のスラックが最小限になるよう選択されます。
- 配線長 : 配線長を最小限に抑えるように、全体的な配置が実行されます。
- 密集 : ピンの密集度が監視され、配線の密集を削減するためセルが分散されます。

配置を開始する前に、デザイン ルール チェック (DRC) (`report_drc` で選択された DRC およびビルトインの内部 DRC) が実行されます。内部 DRC では、次がレポートされます。

- LOC 制約の設定されていない Memory Interface Generator (MIG) セル
- 競合する IOSTANDARD が設定された I/O バンク

DRC の実行後、クロックおよび I/O セルが配置されてからその他のロジックセルが配置されます。クロックと I/O セルは、選択したザイリンクス デバイス ファミリーに特定の複雑な配置規則によって関連していることがよくあるので、同時に配置されます。この時点では、次がターゲットとなります。



- I/O ポートおよびロジック
- グローバルおよびローカル クロック バッファ
- クロック マネージメント タイル (MMCM および PLL)
- ギガビット トランシーバー (GT) セル

固定されていないロジックでは、LOC や AREA\_GROUP/PBLOCK プロパティなどの物理制約に従います。既存の LOC 制約は、ネットリストの接続およびデバイス サイトに対して有効かどうかチェックされます。MIG や GT のような一部の IP は、デバイス専用の配置制約を使用して生成されます。

デバイスの I/O アーキテクチャのため、LOC 制約により LOC 制約が設定されていないセルが制約されることがあります。入力ポートに LOC 制約が設定されている場合、関連する I/O バッファ、IDELAY、ILOGIC の位置も固定されます。競合する LOC 制約は、入力パスの個々のセルには適用できません。出力および GT 関連のセルの場合も同様です。

クロック リソースの配置は、『7 シリーズ FPGA クロック リソース ユーザー ガイド』(UG472) に示す配置規則に従う必要があります。たとえば、グローバル クロック を駆動する入力クロック 兼用 I/O サイトに配置する必要があります。これらのクロック配置規則も、論理 ネットリストの接続およびデバイス サイトに対して有効かどうかチェックされます。

クロックおよび I/O の適切な配置が見つからなかった場合は、違反のあった配置規則が簡単な説明と影響を受けたセルと共に表示されます。まずサイトにセルが暫定的に配置され、その後配置問題を解決するために別のセルが配置されることがあります。暫定的な配置により、クロックおよび I/O 配置のエラーの原因がわかることがよくあります。暫定的な配置でエラーになったセルを手動で配置すると、配置が改善することがあります。



**ヒント** : `place_ports` コマンドを実行してクロックおよび I/O を配置してから、`place_design` コマンドを実行します。ポート配置でエラーが発生した場合、配置はメモリに保存され、エラーを解析できます。

クロックおよび I/O を配置した後、残りの配置プロセスは、グローバル配置、詳細な配置、パッキング、有効化です。配置後、タイミング サマリ (概算) がログ ファイルに出力されます。

```
Phase 12 Placer Reporting
INFO: [Place-100] Post Placement Timing Summary | WNS=-0.08836 | TNS=-1.479 |

説明: WNS = ワースト ネガティブ スラック、TNS = トータル ネガティブ スラック
```



**推奨** : 配置後に `report_timing` を実行し、クリティカル パスをチェックしてください。ネガティブ スラックが大きいパスは、タイミング クロージャを達成するため、手動配置、制約の変更、またはロジックの再構築が必要な場合があります。

## place\_design

ポートとセルを自動配置します。

```
place_design [-effort_level <arg>] [-no_timing_driven] [-quiet] [-verbose]
```

### サンプルスクリプト

```
# Run logic optimization and placement, report utilization and timing estimates,
opt_design
place_design
write_checkpoint -force $outputDir/post_place
report_timing_summary -file $outputDir/post_place_timing_summary.rpt
```

このサンプル スクリプトでは、メモリ内のデザインにロジック最適化を実行し、メモリに書き込み、デザインを配置して、メモリ内のデザインを再び書き込みます。配置後にデザイン チェックポイントを保存し、タイミング サマリ レポートを生成して指定のファイルに記述します。



## 物理合成

物理合成はオプションのプロセスで、ファンアウトが大きいドライバーとクリティカルパスのセルに対してタイミングベースのロジック複製が実行されます。ドライバーが複製され、複製されたドライバーにロードが分散され、それらのドライバーが自動的に配置されます。最適化プロセスは、次のとおりです。

1. 負のスラックが WNS の 10% 以内であるファンアウトの大きいネットは、複製が考慮されます。
2. ロードはその配置によりクラスター化され、各ロード クラスターに対してドライバーが複製および配置されます。
3. タイミングが再解析され、タイミングが改善した場合はロジックの変更が確定されます。
4. 複製後、複製が必要なファンアウトの大きいネットがあるかどうか、デザインが再びチェックされます。ファンアウトの大きいネットがまだある場合、なくなるまで複製プロセスが続行されます。

物理合成では、複製された各ネット、ドライバーの複製回数、最適化前後の最悪の負のスラック (WNS) がレポートされます。複製されたオブジェクトの名前は、元のオブジェクト名に `_replica` と複製されたオブジェクト カウントが付いたものになります。

物理合成では、タイミングが満たされないセルも複製されます。特定のセルのロード同士が離れている場合、セルが複製され、新しいドライバーがロード クラスターの近くに配置されます。この最適化は、パスがワースト ネガティブ スラックの 10% 以内でタイミングを満たしていなければ、ファンアウトが大きくなっても実行されます。

### phys\_opt\_design

ファンアウトの大きいネットのタイミングドリブン複製などの物理最適化を実行し、タイミング結果を向上します。配線済みデザインに対して `phys_opt_design` を実行することはできません。

```
phys_opt_design [-quiet] [-verbose]
```

#### サンプルスクリプト

```
phys_opt_design
```



**重要:** `phys_opt_design` コマンドは、メモリ内のデザインに対して実行されます。2 回実行した場合、1 回目の `run` の結果が最適化されます。

## 配線

Vivado 配線エンジンでは、配置済みデザインに対して配置を実行し、ホールド タイム違反を解決するため配線済みデザインの最適化を実行します。デフォルトでは配線はタイミングドリブンですが、オフにすることもできます。

配線ツールは、次の 2 つモードで実行できます。

- 通常モード：配置済みデザインから開始し、すべてのネットの配線を試みます。配置済みの未配線デザインか、一部またはすべて配線された配線済みデザインのいずれかから開始します。route\_design コマンドはインクリメンタルに実行され、部分的に配線されているデザインに対しては、始めから配線し直すのではなく、既存の配線が開始点として使用されます。これが通常使用されるデフォルト モードです。
- 再配線モード：再配線は、配線を複数回実行するための特別なモードです。再配線モードを使用しない場合、各配線後に配線機能が終了してメモリがクリアされ、配線を実行するごとに配線機能が初期化されます。配線を複数回実行する場合、これは時間がかかります。再配線モードでは、配線が再び実行されることを考慮して、メモリにデータ構造が保持されるので、実行時間を大幅に短縮できます。個別のネットの配線解除および配線などのコマンドは、すぐに実行されます。

配線を開始する前に、デザイン ルール チェック (DRC) (report\_drc で選択された DRC およびビルトインの内部 DRC) が実行されます。

Vivado Design Suite ではまず、クロック、リセット、I/O、専用リソースなど、グローバル リソースが配線されます。このデフォルトの優先順位は配置機能にビルトインされています。その後、タイミングがどれだけクリティカルかに基づいて、データ信号に優先順位が付けられます。配線前のフローや固定配線制約を使用する場合、タイミング制約が適切でないと、一部の信号が最適に配置されなくなることがあるので、注意してください。たとえば、クロックドメインをまたがるパスや、ホールド タイミングで配線遅延が追加される複数サイクルパスなどがその例です。また、密集しているエリアも例として挙げられますが、これは RTL 合成でファンアウトの最適化により解決できます。

最適に配線されていないネットがある場合、問題は多くの場合間違ったタイミング制約です。配線設定を変更する前に、制約が適切であるかどうかを確認してください。配線の前の配置済みデザインのタイミングレポートを参照して、タイミングと制約を確認します。制約を改善したり、RTL の変更を考慮したほうが、配線リソースを追加するよりもよい結果が得られます。

## route\_design

現在のデザインに含まれるネットを配線し、ターゲット パーツでのロジック接続を完成させます。構文は次のとおりです。

```
route_design [-unroute] [-re_entrant <arg>] [-nets <args>] [-physical_nets]
             [-pin <args>] [-effort_level <arg>] [-no_timing_driven] [-preserve] [-delay]
             [-free_resource_mode] -max_delay <arg> -min_delay <arg> [-quiet] [-verbose]
```

### 通常配線のサンプル スクリプト

```
route_design -no_timing_driven -effort_level low
```

通常モードの配線は、インプリメンテーション run の一部として、または Tcl スクリプトの一部として place\_design の後に route\_design コマンドを使用して実行します。配線が終了すると、リソースタイプ別に使用される配線リソースの統計およびタイミング サマリが示されます。

```
[Route-20] Post Routing Timing Summary | WNS=0.0585 | TNS=0 | WHS=0 | THS=0 |
```

説明：

- WNS: ワースト ネガティブ スラック
- TNS: トータル ネガティブ スラック
- WHS: ワースト ホールド スラック

- 。 THS: トータル ホールド スラック

#### 再配線のサンプル スクリプト 1

```
# route a few critical nets
% route_design -delay -nets [get_nets myPreRoutes*]
# Complete full route
% route_design
```

-nets または -pin のような再配線オプションを使用すると再配線モードになり、次のような特定の配線問題を修正するためにインタラクティブに実行されます。

- 完全に配線する前に、クリティカル ネットおよびロック ダウン リソースを事前に配線
- クリティカルではないネットの配線を手動で解除して、クリティカル ネットにより多くの配線リソースを使用できるようにする

最初の再配線コマンドでは、配線機能が初期化されてから、クロックなどの重要なネットが配線され、タイミング解析、タイミングドリブンの配線およびホールド調整が実行されます。再配線モードがオンになった後は、実行された配線および未配線の結果がメモリに保存されます。再配線モードを解除するには、route\_design を実行するか、次のコマンドを使用してオフにします。

```
route_design -re_entrant off
```

#### 再配線のサンプル スクリプト 2

```
% set preRoutes [list]
% foreach net [get_nets -hier] {
    set weight [get_property weight $net]
    if {$weight > 5} { ; # get nets with weight above 5
        lappend preRoutes $net
    }
}
% route_design -nets [get_nets $preRoutes]

# Unroute all the nets in u0/u1, and route the critical nets first
% route_design -unroute [get_nets u0/u1/*]
% route_design -delay -nets [get_nets $myCritNets]
% route_design
```

このサンプル スクリプトでは、ネットの weight プロパティに基づいて preRoutes というネットのリストを作成し、このプロパティの値が大きいネットを最初に配線します。これには、配線前にクリティカル ネットに weight プロパティを設定する必要があります。

**注記:** weight プロパティの使用方法は、『Vivado Design Suite ユーザー ガイド: 制約の使用』(UG903) を参照してください。

ネット ウェイト値の大きいネットを配線し、route\_design が完了すると、u0/u1 インスタンスのネットの配線を解除して、まずクリティカル ネット myCritNets を配線してから、残りの未配線ネットを配線します。

配線中のデザイン解析には、次のコマンドを使用できます。

- report\_route\_status: ネットの配線ステータスをレポートします。
- report\_timing: パス エンドポイント解析を実行します。

すべての Tcl コマンドおよびそのオプションの詳細は、『Vivado Design Suite Tcl コマンド リファレンス ガイド』(UG835) を参照してください。

# リモート ホストの使用

## リモート Linux ホストでの run の起動

Vivado™ IDE では、複数の Linux ホストで同時に合成およびインプリメンテーション run を実行できます。これは、Oracle® 社の Grid Engine や IBM® 社の Platform LSF などの簡易版を使用した機能です。セキュリティおよび Microsoft Windows システムにリモート シェル機能がないため、リモート ホストは Linux でのみサポートされます。

ジョブ投入アルゴリズムは、Linux OS のサービスであるセキュア シェル (SSH) 内の Tcl パイプを使用したラウンドロビン形式でインプリメントされています。Vivado IDE で複数の Linux ホストで run を起動する前に、リモート run を起動するたびにパスワードを入力しなくて済むように SSH を設定する必要があります。SSH の設定方法は、[47 ページの「SSH の設定」](#)を参照してください。

リモート Linux ホストで合成およびインプリメンテーション run を実行する際の要件は、次のとおりです。

- Vivado ツールのインストールがログイン シェルから使用できると想定されるので、\$XILINX および \$PATH 環境変数が .cshrc/.bashrc セットアップ スクリプトで正しく設定されます。リモート マシンにログインし、ほかのスクリプトをソースとせずに「vivado -help」と入力できる場合、このフローは機能します。ログイン時に .cshrc または .bashrc で Vivado が設定されていない場合、[Run pre-launch script] を使用して、環境設定スクリプトをすべてのジョブ前に実行できます。
- Vivado IDE のインストールがリモート マシンの割り当てられたネットワークから表示できるようにする必要があります。Vivado IDE がマシンのローカル ディスクにインストールされている場合は、リモート マシンからは表示できない可能性があります。
- Vivado IDE のプロジェクト ファイル (.xpr) およびディレクトリ (.dita および .runs) が、リモート マシンの割り当てられたネットワークから表示できるようにする必要があります。デザイン データがローカル ディスクに保存されていると、リモート マシンからは表示できない可能性があります。

## リモート ホストの設定

リモート Linux ホストで合成またはインプリメンテーションを実行できるように Vivado IDE を設定するには、次の手順に従います。

1. 次のいずれかのコマンドを選択します。
  - [Tools] → [Options] → [Remote Hosts]
  - [Run Synthesis] → [Launch Runs] → [Configure Hosts]
  - [Implementation] → [Launch Runs] → [Configure Hosts]
  - [Launch Selected Runs] ダイアログ ボックスの [Configure Hosts] ([図 A-1](#))

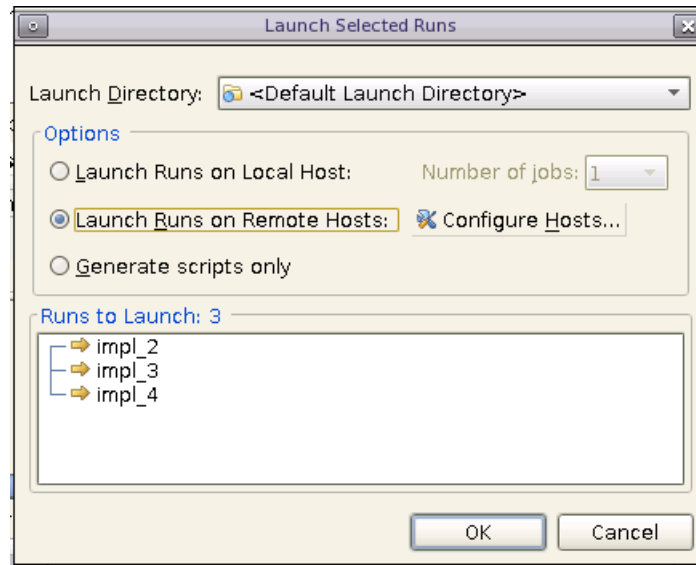


図 A-1 : [Launch Selected Runs] ダイアログ ボックスの [Configure Hosts]

[Vivado Options] ダイアログ ボックスの [Remote Hosts] ページが開き、定義されているリモート Linux ホストが表示されます (46 ページの図 A-2)。

2. [Add] ボタンをクリックし、リモート サーバー名を入力します。
3. [Jobs] 列で、リモート マシンで同時実行に使用できるプロセッサの数を指定します。各 run は個別のプロセッサで実行されます。Vivado IDE では、プロセッサのマルチスレッドはサポートされません。
4. [Enabled] チェック ボックスをオン/オフにして、サーバーを使用するかどうかを指定します。このチェック ボックスは、選択した run にどのサーバーを使用するか指定するのに使用できます。
5. run を実行する際に使用するリモート アクセス コマンドを変更する場合は、[Launch jobs with] で指定します。デフォルトのコマンドは「`ssh -q -o BatchMode=yes`」です。



**重要：**このフィールドを変更する場合は、細心の注意を払ってください。たとえば、`BatchMode =yes` を削除すると、セキュア シェルでパスワードのプロンプトが表示されるため、プロセスが停止してしまいます。

6. run の起動前に実行するスクリプトを定義する場合は、[Run pre-launch script] をオンにします。ログイン時に Vivado IDE が設定されていない場合に、このオプションを使用してホスト環境を設定するスクリプトを実行します。
7. 結果を移動またはコピーするなど、run の完了後に実行するカスタム スクリプトを定義する場合は、[Run post-completion script] をオンにします。
8. run の完了時に電子メールが送信されるようにする場合は、[Send email to] をオンにします。各ジョブの後に通知を送信するか ([After each Job])、すべてのジョブの完了後に通知を送信するか ([After all jobs]) を選択できます。
9. 設定が終了したら、[OK] をクリックします。

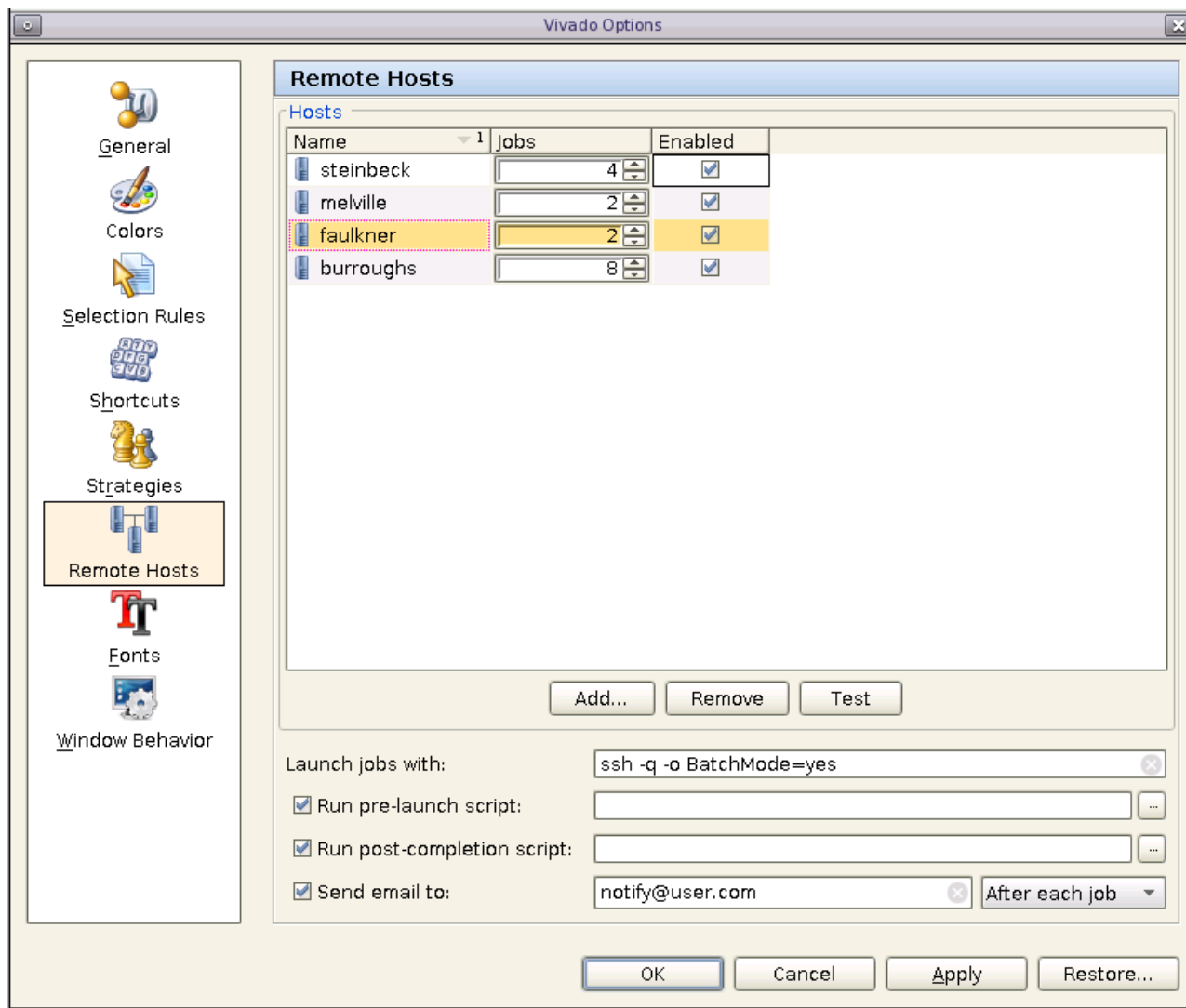


図 A-2: リモート ホストの設定

1 つまたは複数のホストを選択し、[Test] をクリックすると、リモート ホストへの接続をテストし、そのサーバーが使用可能かどうか、コンフィギュレーションが正しく設定されているかを確認できます。



**推奨:** ホストで run を実行する前に、各ホストをテストして正しく設定されていることを確認してください。

リモート ホストを削除するには、ホストを選択して [Remove] をクリックします。

## SSH の設定

SSH は、Linux ターミナルまたはシェルで次のコマンドを入力して設定します。

**注記：**これは一度設定しておけば、繰り返し設定する必要はありません。

1. Linux ターミナルまたはシェルで次のコマンドを実行し、プライマリ コンピューターでパブリック キーを生成します。必須ではありませんが、セキュリティ保護のため、プライベート キーを入力し、記憶しておくことをお勧めします。

```
ssh-keygen -t rsa
```

2. パブリック キーをリモート マシンの `authorized_keys` ファイルに追加します。`remote_server` をホスト名に変更します。

```
cat ~/.ssh/id_rsa.pub | ssh remote_server "cat - >> ~/.ssh/authorized_keys"
```

3. 次のコマンドを実行して、プライベート キーのパスフレーズの入力をプロンプトし、キー転送を有効にします。

```
ssh -add
```

これで、どのリモート マシンでもパスワードを入力せずに使用できます。新しいマシンに初めてアクセスする場合は、パスワードを入力するよう求められますが、次回からは入力する必要はありません。毎回パスワードの入力を求められる場合は、システム管理者に連絡してください。

## その他のリソース

---

### ザイリンクス リソース

アンサー、資料、ダウンロード、フォーラムなどのサポート リソースは、次のザイリンクス サポート サイトを参照してください。

<http://japan.xilinx.com/support>

ザイリンクス資料で使用する用語集は、次を参照してください。

<http://japan.xilinx.com/company/terms.htm>

---

### ソリューション センター

デバイス、ツール、IP のサポートについては、[ザイリンクス ソリューション センター](#)を参照してください。トピックには、デザイン アシスタント、アドバイザリ、トラブルシュート ヒントなどが含まれます。

---

### 参考資料

Vivado™ Design Suite 2012.2 資料ページ : [http://japan.xilinx.com/support/documentation/dt\\_vivado\\_vivado2012-2.htm](http://japan.xilinx.com/support/documentation/dt_vivado_vivado2012-2.htm)