# コマンド ラインでの SmartXplorer の使用

## チュートリアル (ISE 12.3)

UG688 2010 年 9 月 21 日

XILINX®

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

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

© 2010 Xilinx, Inc. XILINX, the Xilinx logo, Virtex, Spartan, ISE, 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 (v1.3) を翻訳したもので、内容に相違が生じる場合には原文を優先します。
資料によっては英語版の更新に対応していないものがあります。
日本語版は参考用としてご使用の上、最新情報につきましては、必ず最新英語版をご参照ください。

# 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 |
|---|---|---|
| 2009 年 6 月 12 日 | 1.0 | 初期リリース |
| 2009 年 7 月 20 日 | 1.1 | 11.2 用にアップデート |
| 2010 年 4 月 19 日 | 1.2 | 12.1 用にアップデート |
| 2010 年 9 月 7 日 | 1.3 | 12.3 用にアップデート |

# 目次

# このチュートリアルについて

このチュートリアルでは、SmartXplorer の概要と、その機能をタイミング クロージャを達成するために使用する方法を示します。

このチュートリアルには、2 つのバージョンがあります。

- ISE® Project Navigator ユーザー用 :
  SmartXplorer を Project Navigator から使用する方法を説明します。
- コマンド ライン ユーザー用 :
  ほとんどの Microsoft Windows ユーザーは ISE Project Navigator からザイリンクス ツールを使用しているので、このチュートリアルでは主に Linux プラットフォームでの SmartXplorer の使用について説明します。内容は、必要に応じて Microsoft Windows オペレーティング システムにも簡単に適用できます。

どちらのバージョンにも、詳細な手順を示す演習が含まれており、SmartXplorer のさまざまな機能を順を追って学ぶことができます。

12.1 リリースから、SmartXplorer で Xilinx® Synthesis Technology (XST) および Synplify 合成ツールがサポートされるようになりました。複数のインプリメンテーション ストラテジを実行する前に、複数の合成ストラテジを実行してインプリメンテーションの実行に使用する最良の合成済みネットリストを選択できます。SmartXplorer で合成を実行する必要はありません。これまでと同様、インプリメンテーションのみに SmartXplorer を使用することも可能です。このチュートリアルでは、SmartXplorer をインプリメンテーションでのみ使用するフローについても説明します。

Synplify サポートについては、『コマンド ライン ツール ユーザー ガイド』(UG628) を参照してください。

**メモ** : SmartXplorer での合成の実行はコマンド ラインのみでサポートされ、ISE 環境では使用できません。このチュートリアルでは、SmartXplorer のすべてのオプションおよび機能は説明していません。

## チュートリアルの内容

このチュートリアルは、次の章から構成されています。

- 第 1 章「SmartXplorer の概要」: SmartXplorer の機能を説明します。
- 第 2 章「チュートリアルの説明」: 各演習で学ぶ機能について概要を説明します。各演習を完了するのに必要な時間も示します。
- 第 3 章「演習の準備」: 各演習で使用するデザインの入手先とインストール方法を示します。

- 第 4、5、6 章 : 各演習の詳細を示します。演習の概要と、演習を完了するために必要な詳細な手順を説明します。
  - 第 4 章「演習 1 : 基本フロー」
  - 第 5 章「演習 2 : カスタム ストラテジの作成」
  - 第 6 章「演習 3 : 複数のストラテジの同時実行」
- 付録 A「カスタム ファイル」: カスタム ストラテジ ファイルおよびホスト リスト ファイルの例を示します。また、さまざまなタスクでの SmartXplorer の設定も含まれています。
- 付録 B「XST のスクリプト ファイルの作成」: SmartXplorer で XST 合成を実行するための XST スクリプト ファイルの作成方法を示します。

# その他のリソース

追加資料は、次のウェブサイトを参照してください。

http://japan.xilinx.com/support/documentation/index.htm

シリコン、ソフトウェア、IP に関する問題をアンサー データベースで検索したり、テクニカル サポートのウェブ ケースを開くには、次のウェブサイトにアクセスしてください。

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

# 表記規則

このマニュアルでは、次の表記規則を使用しています。各規則について、例を挙げて説明します。

## 書体

次の規則は、すべてのマニュアルで使用されています。

| 表記規則 | 使用箇所 | 例 |
|---|---|---|
| `Courier フォント` | システムにより表示されるメッセージ、プロンプト、プログラム ファイルを表示します。 | `speed grade:- 100` |
| **`Courier フォント (太字)`** | 構文内で入力するコマンドを示します。 | **`ngdbuild`** *design_name* |
| イタリック フォント | ユーザーが値を入力する必要のある構文内の変数に使用します。 | **`ngdbuild`** *design_name* |
| 二重/一重かぎかっこ 『』、「」 | 『』はマニュアル名を、「」はセクション名を示します。 | 詳細は、『コマンド ライン ツール ユーザー ガイド』の「PAR」を参照してください。 |
| 角かっこ [ ] | オプションの入力またはパラメータを示します。**`bus[7:0]`** のようなバス仕様では必ず使用します。 | **`ngdbuild`** [*option_name*] *design_name* |
| | GUI 表記に使用します。 | [File] → [Open] をクリックします。 |

| 表記規則 | 使用箇所 | 例 |
|---|---|---|
| 中かっこ { } | 構文内で、複数の項目から 1 つ以上を選択する必要がある場合に使用します。 | **lowpwr ={on\|off}** |
| 縦棒 \| | 構文内で、選択可能な複数の項目を分離するために使用します。 | **lowpwr ={on\|off}** |
| 縦の省略記号<br>.<br>.<br>. | 繰り返し項目が省略されていることを示します。 | IOB #1:Name = QOUT'<br>IOB #2:Name = CLKIN'<br>.<br>.<br>. |
| 横の省略記号 **...** | 繰り返し項目が省略されていることを示します。 | **allow block** *block_name loc1 loc2 ... locn;* |

## オンライン マニュアル

このマニュアルでは、次の規則が使用されています。

| 表記規則 | 使用箇所 | 例 |
|---|---|---|
| 青色の文字 | マニュアル内の相互参照を示します。 | 詳細は、「その他のリソース」を参照してください。<br>詳細は、第 1 章「タイトル フォーマット」を参照してください。 |
| 青色の下線付き文字 | ウェブサイト (URL) へのハイパーリンクを示します。 | 最新のスピード ファイルは、http://japan.xilinx.com から入手できます。 |

XILINX®

第 1 章

# SmartXplorer の概要

## 概要

SmartXplorer は、最短時間でタイミング クロージャを達成することを目標としています。

タイミング クロージャは、FPGA デザインにおいて最大の課題の 1 つです。ザイリンクスでは、タイミング クロージャの達成をサポートするため、次の点に焦点を置いています。

- 合成およびインプリメンテーション アルゴリズムの向上
- PlanAhead™ ソフトウェアおよび FPGA Editor などの高度なグラフィカル解析ツールの提供

FPGA ツールは使いやすくなってきており、高度な機能を提供するようになってきていますが、すべてのデザイン状況を予測するのは困難です。デザイン サイクルの最終段階になってから問題が見つかる場合もあります。

設計者の経験レベルにかかわらず、HDL コードを変更したり配置制約を使用する前に、異なるツール オプションを変更するのが通常のアプローチです。ツール オプションを変更するのは非常に簡単ですが、よりよいオプションをどのように見つけ出すかが課題となります。

## SmartXplorer の主な利点

SmartXplorer には、主に次の 2 つの利点があります。

- ビルトインまたはカスタム インプリメンテーション ストラテジを複数使用して、タイミングを満たすようにデザインを実行します。

  **メモ**：デザイン ストラテジとは、エリア、スピード、消費電力など、特定のデザインの目標を達成するためのプロセス プロパティとその値の組み合わせです。
- 複数のストラテジを複数のマシンで同時に実行し、ジョブを短時間で完了します。

### デザイン ストラテジ

SmartXplorer では、あらかじめ定義されたビルトイン ストラテジが複数提供されています。これらのストラテジは、各 FPGA ファミリ用に個別に選択され、調整されています。この選択は、特定のソフトウェア バージョンで最適に機能することを確実にするため、各メジャー リリースで修正されます。

独自の経験に基づいて、カスタム ストラテジまたはスクリプトを作成することもできます。SmartXplorer ではこれらのカスタム ストラテジをシステムに統合し、それらのみを使用したり、SmartXplorer で提供されているストラテジと組み合わせて使用できます。

SmartXplorer は、プロジェクトの最終段階だけでなく、プロジェクト サイクル全体で使用することにより、デザイン サイクルの最終段階で緊急事態が発生するのを回避または削減できます。定期的に実行してデザインの状態を監視し、タイミングが許容範囲内にあることを確認することをお勧めします。

## 並列実行

複数のデザイン ストラテジ (ジョブ) を同時に実行すると、プロジェクトを短時間で完了できます。この機能は、ご使用のオペレーティング システムによって異なります。

### Linux OS

ネットワーク上の複数のマシンで複数のジョブを並列実行できます。これには、次の 2 つの方法があります。

- 通常の Linux ネットワーク : ネットワーク上でのジョブの分配は SmartXplorer で管理されます。使用するマシンのリストが必要です。
- LSF (Load Sharing Facility) または SGE (Sun Grid Engine) コンピューター ファーム : ジョブの分配は、LSF または SGE で管理されます。SmartXplorer に同時に割り当て可能なマシンの数を指定します。

Linux ネットワークへのアクセスがない場合でも、個人の Linux マシンにマルチコア プロセッサまたは複数のプロセッサがあれば、複数のジョブを並列実行できます。

### Microsoft Windows OS

1 つの Windows マシンにマルチコア プロセッサまたは複数のプロセッサが搭載されている場合、複数のストラテジを同時に実行できます。

## 1 つの Linux または Windows マシンの使用

ネットワーク上の Linux サーバーへのアクセスがなく、ローカル コンピューターしか使用できない場合は、マシンにマルチコア プロセッサがあるか複数のプロセッサがあることを確認してください。

まず、ご使用のマシンで同時に実行可能なジョブの数を予測する必要があります。

論理的には、同時に実行可能なジョブの数は次のように算出されます。

```
Nb_Of_Jobs = P * C
```

ここで、P はプロセッサの数、C はプロセッサごとのコアの数です。

たとえば、デュアル コア プロセッサが 4 つある場合、8 個のジョブを同時に実行できます。

ただし、使用可能なメモリ、メモリの速度、ハード ドライブの速度によっては、コンピューターで上記の式で求めた最大数のジョブを処理できない可能性があります。

次に、並列実行に関するヒントを示します。

**ヒント 1** : デザインのメモリ要件のため、マシンで一度に 1 つのストラテジしか実行できない場合は、すべてのストラテジを順次に実行する必要があります。このような場合、夜間に SmartXplorer を実行すると有益です。

**ヒント 2** : タイミングの問題を解決している場合、デザインに含まれるブロックを個別に実行することもできます。これらのブロックに対してであれば複数のストラテジを同時に実行できる可能性があり、時間を節約できます。

XILINX®

第 2 章

# チュートリアルの説明

このチュートリアルでは小型のデザインを使用しており、演習を短時間で完了できるようになっています。SmartXplorer の主な機能をすべて含むチュートリアル全体を、40 分以内で完了できます。

演習は、順番に実行することをお勧めします (演習 1、演習 2、演習 3 など)。ただし、各演習は独立しており、特定の機能をすぐに学ぶ必要がある場合は、どの順序で実行しても問題ありません。

SmartXplorer には、主に次の 2 つの利点があります。

1. あらかじめ定義されたビルトイン デザイン ストラテジまたはユーザー定義のカスタム ストラテジを使用することにより、タイミング クロージャの達成をサポートします。
2. 複数のストラテジを複数のマシンで同時に実行し、ジョブを短時間で完了します。

このチュートリアルでは、これらの 2 つの機能を別々に説明します。

- 演習 1 および演習 2 では、タイミング クロージャについて説明します。これら 2 つの演習では、すべてのデザイン ストラテジを 1 つずつ順番に実行します (複数のストラテジを並列実行する機能はオフ)。
- 演習 3 では、複数のストラテジの並列実行について説明します。

次の表に、すべての演習の概要を示します。

**表 2-1 : 演習の概要**

| タイトル | 時間 | 説明する機能 |
|---|---|---|
| 演習 1 : 基本フロー | 25 分 | • SmartXplorer の起動方法 (合成およびインプリメンテーション)<br>• 最終結果のレポート方法、保存方法、および活用方法<br>• 前回の実行で得られた結果を向上する方法<br>• ビルトイン ストラテジと追加の反復を一度に実行するよう SmartXplorer を設定する方法 (夜間実行など)<br>• SmartXplorer をインプリメンテーション フローのみで実行する方法 |
| 演習 2 : カスタム ストラテジの作成 | 5 分 | • カスタム ストラテジ ファイルを作成し、SmartXplorer で使用する方法 |
| 演習 3 : 複数のストラテジの同時実行 | 10 分 | • 複数のストラテジの並列実行 |

## 演習の実行に必要な知識

演習を実行するには、ザイリンクス FPGA インプリメンテーション フローの主要なステップ (合成、変換 (NGDBuild)、マップ、配置配線、およびタイミング解析 (TRCE)) とその実行方法に関する次の基本的な知識が必要です。

第 3 章

# 演習の準備

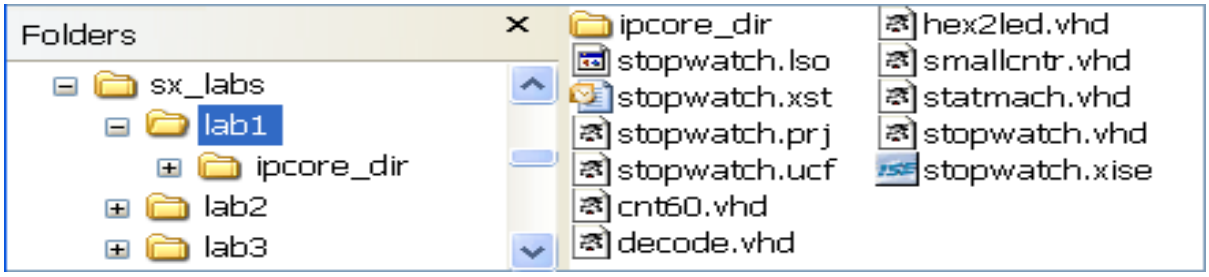
この章では、各演習用にデザインを準備する方法を説明します。

## デザインの概要

このチュートリアルのすべての演習で、Spartan®-6 xc6slx4-3-tqg144 デバイスをターゲットとする小型の stopwatch デザインを使用します。

## 手順

1. 次のサイトから UG688.zip ファイルをダウンロードします。
   https://secure.xilinx.com/webreg/clickthrough.do?cid=132685&license=RefDesLicense
2. SmartXplorer 演習データを保存する sx_labs ディレクトリを作成します。
3. UG688.zip ファイルを sx_labs ディレクトリにコピーします。
4. UG688.zip ファイルを解凍します。ディレクトリ構造は、次のようになります。



XILINX®

第 4 章

# 演習 1：基本フロー

## 目標

この演習では、次の内容を学びます。

- ビルトイン SmartXplorer ストラテジの実行方法 (合成およびインプリメンテーション)
- 最終結果のレポート方法、保存方法、および活用方法
- 前回の実行で得られた結果に基づき、実行を追加で反復してタイミングを向上する方法
- SmartXplorer をインプリメンテーション フローのみで実行する方法

## 背景

12.1 リリースから、SmartXplorer で Xilinx® Synthesis Technology (XST) および Synplify 合成ツールがサポートされるようになりました。複数のインプリメンテーション ストラテジを実行する前に、複数の合成ストラテジを実行してインプリメンテーションの実行に使用する最良の合成済みネットリストを選択できます。SmartXplorer で合成を実行する必要はありません。これまでと同様、インプリメンテーションのみに SmartXplorer を使用することも可能です。

**重要**：このチュートリアルは、次のように構成されています。

- まず、合成 (XST) とインプリメンテーション フローの両方を SmartXplorer で実行します。
- SmartXplorer をインプリメンテーション フローのみで使用する方法も説明します。

SmartXplorer で合成を実行する場合、合成とインプリメンテーションの 2 段階プロセスになります。

- 段階 1 (合成)：複数の合成ストラテジを実行し、パフォーマンスが最良のネットリストを特定します。合成ツールではタイミング スコアは生成されません。合成された各ネットリストに対して、ランタイムが最短になるよう最適化されたストラテジ (クイック インプリメンテーション) を使用して MAP および PAR を 1 回実行し、タイミング スコアを算出します。

**メモ**：クイック インプリメンテーション ストラテジは、インプリメンテーション用のビルトイン SmartXplorer ストラテジの 1 つです。

- 段階 2 (インプリメンテーション)：最良のネットリストを使用し、複数のインプリメンテーション ストラテジを実行してタイミング要件が満たされるようにします。

SmartXplorer をインプリメンテーション フローのみに使用する場合は、段階 2 のみが実行されます。

# 演習

## 手順 1 : ザイリンクス環境の設定

1. ターミナル ウィンドウを開き、ザイリンクス環境を設定します。
2. **cd** コマンドを使用して lab1 ディレクトリに移動します。

## 手順 2 : 必須オプションを使用した SmartXplorer の実行

SmartXplorer (**smartxplorer**) を実行するには、次のオプションを指定する必要があります。

- ターゲット デバイス : **-p xc6slx4-3-tqg144**
- UCF 制約ファイル : **-uc stopwatch.ucf**
- XST スクリプト ファイル : stopwatch.xst

**メモ** : XST スクリプト ファイルの作成については、付録 B で説明します。

このチュートリアルでは、これらに加え、次の 2 つのオプションも使用します。

- チュートリアル デザインには、ipcore_dir ディレクトリに外部コア (tenth.ngc) が含まれているので、**-sd** オプションを使用してこのディレクトリを参照します。
- **-wd smartxplorer_results** オプションを使用して、SmartXplorer の結果を smartxplorer_results というディレクトリに保存します。

これで、SmartXplorer を実行する準備ができました。

1. ターミナル ウィンドウに次のコマンドを入力して SmartXplorer を起動します。

```
smartxplorer -p xc6slx4-3-tqg144 -uc stopwatch.ucf
-wd smartxplorer_results -sd ipcore_dir stopwatch.xst
```

SmartXplorer の実行を開始すると、ステータスの表に進行状況が示され、最終的な結果のサマリが表示されます。この表の各行が、1 つの SmartXplorer ストラテジを表します。この表は、次の場所に表示されます。

- ターミナル ウィンドウ
- smartxplorer_results ディレクトリの smartxplorer.html ファイル。このファイルは、ウェブ ブラウザーで開いてください。

この表は、SmartXplorer の実行中随時アップデートされます。次に、smartxplorer.html の中間状態の例を示します。

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| XSTOptReshRedcon_MapRunTime | host_1 | run1 | **Done** | 2118 | 30 (1%) | 29 (1%) | 0h 1m 5s |
| XSTOptOnehot_MapRunTime | host_1 | run2 | Mapping | None | None | None | 0h 0m 14s |
| XSTOnehot_MapRunTime | None | None | None | None | None | None | None |
| XSTOptOnehotRedcon_MapRunTime | None | None | None | None | None | None | None |
| XSTRegbalOptOnehot_MapRunTime | None | None | None | None | None | None | None |
| XSTOnehotRedcon_MapRunTime | None | None | None | None | None | None | None |
| XSTRegbalOptOnehotReshRedcon_MapRunTime | None | None | None | None | None | None | None |

合成段階では、すべての合成ストラテジ (7 個) とタイミング スコアを算出するために使用されるクイック インプリメンテーション ストラテジが表示されます。

**メモ**：1 番左の列に示されるストラテジ名では、合成ストラテジ名とクイック インプリメンテーション ストラテジ名がアンダースコア (_) でつなげられています。たとえば、XSTOptReshRedcon_MapRunTime は XSTOptReshRedcon という合成ストラテジとクイック インプリメンテーション ストラテジである MapRunTime を表します。

XSTOptReshRedcon_MapRunTime ストラテジの [Status] 列は [Done] となっており、完了していることがわかります。このストラテジの実行には 1 分 5 秒かかり、タイミング スコアは 2118 であるのでタイミング制約は満たされていません。この行は緑色で表示されており、現時点ではこのストラテジで最良のタイミングが得られていることを示します。

MapOnehot_MapRunTime ストラテジは現在実行中で、マップ段階です。

**メモ**：すべてのストラテジの実行を完了する前に SmartXplorer を停止するには、Ctrl + C キーを押します。

すべての合成ストラテジの実行が完了すると、タイミング スコアに基づいて最高のネットリストが選択され、そのネットリストがクイック インプリメンテーション ストラテジ以外のインプリメンテーション ストラテジを使用して実行されます。HTML は次のようにアップデートされます。

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| XSTOptReshRedcon_MapRunTime | host_1 | run1 | **Done** | 2118 | 30 (1%) | 29 (1%) | 0h 1m 5s |
| XSTOptOnehot_MapRunTime | host_1 | run2 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehot_MapRunTime | host_1 | run3 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 59s |
| XSTOptOnehotRedcon_MapRunTime | host_1 | run4 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 50s |
| XSTRegbalOptOnehot_MapRunTime | host_1 | run5 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehotRedcon_MapRunTime | host_1 | run6 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 49s |
| XSTRegbalOptOnehotReshRedcon_MapRunTime | host_1 | run7 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDup | host_1 | run8 | **Done** | 34 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptIOReg | host_1 | run9 | Mapping | None | None | None | 0h 0m 11s |
| XSTOnehotRedcon_MapRegDup | None | None | None | None | None | None | None |
| XSTOnehotRedcon_MapExtraEffortIOReg | None | None | None | None | None | None | None |
| XSTOnehotRedcon_MapLogOptRegDup | None | None | None | None | None | None | None |
| XSTOnehotRedcon_MapExtraEffort2 | None | None | None | None | None | None | None |

この例では、タイミング スコアとランタイムに基づいて、XSTOnehotRedcon 合成ストラテジ (run6) が最良のネットリストとして選択されます。このネットリストを使用して、6 つのインプリメンテーション ストラテジ (run8 ～ run13) が実行されます。すべてのストラテジの実行が完了すると、最良のストラテジが緑色にハイライトされます (run8)。

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| XSTOptReshRedcon_MapRunTime | host_1 | run1 | **Done** | 2118 | 30 (1%) | 29 (1%) | 0h 1m 5s |
| XSTOptOnehot_MapRunTime | host_1 | run2 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehot_MapRunTime | host_1 | run3 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 59s |
| XSTOptOnehotRedcon_MapRunTime | host_1 | run4 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 50s |
| XSTRegbalOptOnehot_MapRunTime | host_1 | run5 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehotRedcon_MapRunTime | host_1 | run6 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 49s |
| XSTRegbalOptOnehotReshRedcon_MapRunTime | host_1 | run7 | **Done** | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDup | host_1 | run8 | **Done** | 34 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptIOReg | host_1 | run9 | **Done** | 5784 | 48 (2%) | 22 (1%) | 0h 0m 45s |
| XSTOnehotRedcon_MapRegDup | host_1 | run10 | **Done** | 561 | 30 (1%) | 32 (1%) | 0h 0m 39s |
| XSTOnehotRedcon_MapExtraEffortIOReg | host_1 | run11 | **Done** | 4283 | 30 (1%) | 22 (1%) | 0h 0m 39s |
| XSTOnehotRedcon_MapLogOptRegDup | host_1 | run12 | **Done** | 701 | 30 (1%) | 32 (1%) | 0h 0m 39s |
| XSTOnehotRedcon_MapExtraEffort2 | host_1 | run13 | **Done** | 784 | 30 (1%) | 32 (1%) | 0h 0m 39s |

2. XSTOptOnehot_MapRunTime ストラテジの [Output] 列の [run2] リンクをクリックすると、ストラテジ ログ ファイル stopwatch_sx.log が開きます。このファイルには、XST、MAP、PAR、および TRCE レポートが含まれます。

```
########################################
WARNING: p,opt_level,fsm_encoding option(s) in xst file will be overridden
by the new values provided in strategy XSTOptOnehot_MapRunTime.
########################################


Command Line :
-------------
xst C:\sx_labs\lab1\smartxplorer_results\run2\stopwatch.xst
-ifn C:\sx_labs\lab1\smartxplorer_results\run2\stopwatch.xst

Release 12.2 - xst M.63c (nt)
Copyright (c) 1995-2010 Xilinx, Inc.  All rights reserved.
--> Parameter TMPDIR set to .

TABLE OF CONTENTS
  1) Synthesis Options Summary
  2) HDL Parsing
  3) HDL Elaboration
  4) HDL Synthesis
       4.1) HDL Synthesis Report
```

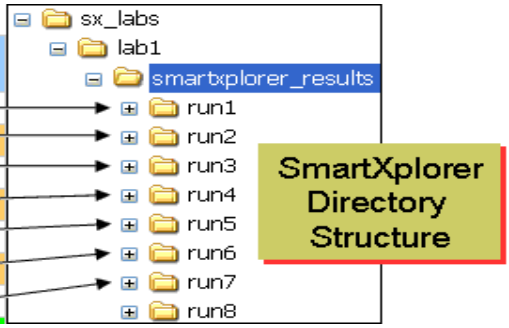
3. [Timing Score] 列の [1895] リンクをクリックすると、タイミング レポート サマリが表示されます。

**Asterisk (*) preceding a constraint indicates it was not met.**
**This may be due to a setup or hold violation.**

| Constraint | Check | Worst Case Slack | Best Case Achievable | Timing Errors | Timing Score |
|---|---|---|---|---|---|
| TS_CLK = PERIOD TIMEGRP "CLK" 2.3 ns HIGH 50% | SETUP | -0.387 | 2.687 | 14 | 1895 |
| | HOLD | 0.266 | | 0 | 0 |

4. 結果のディレクトリ構造を確認します。SmartXplorer の結果は、`smartxplorer_results` ディレクトリに保存されます。各 SmartXplorer ストラテジの結果は、個別のディレクトリ (`run1`、`run2` など) に保存されます。

| Strategy | Host | Output |
|---|---|---|
| XSTOptReshRedcon_MapRunTime | host_1 | run1 |
| XSTOptOnehot_MapRunTime | host_1 | run2 |
| XSTOnehot_MapRunTime | host_1 | run3 |
| XSTOptOnehotRedcon_MapRunTime | host_1 | run4 |
| XSTRegbalOptOnehot_MapRunTime | host_1 | run5 |
| XSTOnehotRedcon_MapRunTime | host_1 | run6 |
| XSTRegbalOptOnehotReshRedcon_MapRunTime | host_1 | run7 |




**重要**：SmartXplorer を再実行するには、SmartXplorer の結果に対して開いているレポートをすべて閉じる必要があります。SmartXplorer が起動する前に、以前に生成されたデータ ファイルはすべて削除されます。開いているレポートがあると、SmartXplorer で以前に生成された結果をクリーンアップできません。この場合、ターミナル ウィンドウに「`permission denied`」というエラーが表示され、SmartXplorer が停止します。

```
ERROR: SmartXplorer could not clean up directory run1 from previous run. Please
ensure that no other programs are using this directory.
```

## 手順 3：実行を追加で反復してタイミングを向上

前の手順で、XSTOnehotRedcon_MapGlobOptLogOptRegDup ストラテジで最良のタイミング結果が得られました。このストラテジを異なるコスト テーブルを使用してさらに 5 回実行し (インプリメンテーション フローのみ)、タイミングを向上します。コスト テーブルの詳細は、『コマンド ライン ツール ユーザー ガイド』を参照してください。

1. XSTOnehotRedcon_MapGlobOptLogOptRegDup ストラテジを追加で 5 回実行するには、**`-vp 5`** オプションを使用します。
2. XSTOnehotRedcon_MapGlobOptLogOptRegDup ストラテジからの MAP および PAR オプションを **`-mo`** および **`-po`** オプションを使用して指定する必要があります。これらのオプションは、`smartxplorer_results` ディレクトリの `smartxplorer.txt` ファイルに記述されています。このファイルには、前回実行されたストラテジのオプションがすべて含まれます。

```
----------------------------------------------------------------------
Strategy :XSTOnehotRedcon_MapGlobOptLogOptRegDup
----------------------------------------------------------------------

Run index    : run8
Xst options :-fsm_encoding one-hot -reduce_control_sets auto

Map Options :-ol high -xe n -global_opt speed -logic_opt on
-register_duplication on -w
Par options :-ol high -xe n
```

XSTOnehotRedcon_MapGlobOptLogOptRegDup ストラテジはデフォルトのコスト テーブル 1 で実行されているので、開始コスト テーブルを 2 (**-t 2**) に設定します。これらのオプションは、MAP および PAR のコマンドに追加する必要があります。

```
-mo "-timing -ol high -xe n -global_opt speed -logic_opt
     -register_duplication on -w -t 2"
-po "-ol high -xe n"
```

3. インプリメンテーション フローを追加で 5 回実行するには、XST スクリプト (`stopwatch.xst`) ではなく合成済みネットリスト (`stopwatch.ngc`) を SmartXplorer の入力ファイルとして使用します。

   次のコマンドを使用して、`stopwatch.ngc` を `smartxplorer_results/run6` から現在のディレクトリ (`lab1`) にコピーします。

   ```
   cp smartxplorer_results/run6/stopwatch.ngc .
   ```

4. ターミナル ウィンドウに次のコマンドを入力して SmartXplorer を起動します。

   ```
   smartxplorer -p xc6slx4-3-tqg144 -uc stopwatch.ucf -wd
   smartxplorer_results -sd ipcore_dir -vp 5 -mo "-ol high -xe n
   -global_opt speed -logic_opt on -register_duplication on -w -t 2" -po
   "-ol high -xe n" stopwatch.ngc
   ```

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| MapRunTime* | host_1 | run1 | Done | 129 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| MapRunTimeCT3* | host_1 | run2 | Done | 0 | 48 (2%) | 32 (1%) | 0h 0m 44s |

ストラテジは 2 回しか実行されませんでした。これは、タイミング制約が満たされた時点で SmartXplorer の実行を終了するようデフォルトで設定されているからです。コスト テーブル 3 (MapRunTimeCT3) でタイミングが満たされたため、その時点で SmartXplorer の実行が終了しました。**-ra** オプションを使用すると、タイミングが満たされた場合でも指定回数だけ実行が反復され、各コスト テーブルのパフォーマンスを比較できます。

## 手順 4 : ビルトイン ストラテジと追加の反復を一度に実行

前の手順では、2 段階でタイミング制約を満たすことができました。最初の段階「手順 2 : 必須オプションを使用した SmartXplorer の実行」でビルトイン ストラテジをすべて実行して最良のストラテジを見つけ、次の段階「手順 3 : 実行を追加で反復してタイミングを向上」で最良のストラテジを異なるコスト テーブルを使用して実行しました。

同じ手順を、SmartXplorer を 1 回起動するだけで実行できます。これは、夜間にジョブを実行する場合などに有益です。

1. 最大実行回数を 18 (ビルトイン ストラテジ 13 個 + 5 個の追加実行) に設定します (**-m 18**)。
2. **-ra** オプションを使用して、タイミングが満たされたかどうかにかかわらず、指定回数だけ実行を反復します。
3. ターミナル ウィンドウに次のコマンドを入力して SmartXplorer を起動します。

   ```
   smartxplorer -p xc6slx4-3-tqg144 -uc stopwatch.ucf
   -wd smartxplorer_results -sd ipcore_dir -m 18 -ra stopwatch.xst
   ```

次のような結果が表示されます。

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| XSTOptReshRedcon_MapRunTime | host_1 | run1 | Done | 2118 | 30 (1%) | 29 (1%) | 0h 1m 19s |
| XSTOptOnehot_MapRunTime | host_1 | run2 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehot_MapRunTime | host_1 | run3 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 49s |
| XSTOptOnehotRedcon_MapRunTime | host_1 | run4 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTRegbalOptOnehot_MapRunTime | host_1 | run5 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehotRedcon_MapRunTime | host_1 | run6 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 49s |
| XSTRegbalOptOnehotReshRedcon_MapRunTime | host_1 | run7 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 55s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDup | host_1 | run8 | Done | 34 | 48 (2%) | 32 (1%) | 0h 0m 49s |
| XSTOnehotRedcon_MapGlobOptIOReg | host_1 | run9 | Done | 5784 | 48 (2%) | 22 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapRegDup | host_1 | run10 | Done | 561 | 30 (1%) | 32 (1%) | 0h 0m 40s |
| XSTOnehotRedcon_MapExtraEffortIOReg | host_1 | run11 | Done | 4283 | 30 (1%) | 22 (1%) | 0h 0m 40s |
| XSTOnehotRedcon_MapLogOptRegDup | host_1 | run12 | Done | 701 | 30 (1%) | 32 (1%) | 0h 0m 40s |
| XSTOnehotRedcon_MapExtraEffort2 | host_1 | run13 | Done | 784 | 30 (1%) | 32 (1%) | 0h 0m 40s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDupCT2 | host_1 | run14 | Done | 129 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDupCT3 | host_1 | run15 | Done | 0 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDupCT4 | host_1 | run16 | Done | 0 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDupCT5 | host_1 | run17 | Done | 0 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDupCT6 | host_1 | run18 | Done | 0 | 48 (2%) | 32 (1%) | 0h 0m 44s |

# 手順 5 : SmartXplorer をインプリメンテーション フローのみで実行

先ほど述べたとおり、SmartXplorer で合成を実行する必要はありません。SmartXplorer をインプリメンテーションのみに使用することも可能です。この手順では、7 個のビルトイン インプリメンテーション ストラテジを実行する方法を示します。

1. インプリメンテーション フローのみを実行するには、XST スクリプト (`stopwatch.xst`) ではなく合成済みネットリスト (`stopwatch.ngc`) を SmartXplorer の入力ファイルとして使用します。次のコマンドを使用して、`stopwatch.ngc` を `smartxplorer_results/run6` から現在のディレクトリ (`lab1`) にコピーします。

   ```
   cp smartxplorer_results/run6/stopwatch.ngc
   ```

2. ターミナル ウィンドウに次のコマンドを入力して SmartXplorer を起動します。

   ```
   smartxplorer -p xc6slx4-3-tqg144 -uc stopwatch.ucf -wd
   smartxplorer_results -sd ipcore_dir stopwatch.ngc
   ```

次のような結果が表示されます。

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| MapRunTime | host_1 | run1 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 1m 0s |
| MapGlobOptLogOptRegDup | host_1 | run2 | Done | 34 | 48 (2%) | 32 (1%) | 0h 0m 50s |
| MapGlobOptIOReg | host_1 | run3 | Done | 5784 | 48 (2%) | 22 (1%) | 0h 0m 44s |
| MapRegDup | host_1 | run4 | Done | 561 | 30 (1%) | 32 (1%) | 0h 0m 40s |
| MapExtraEffortIOReg | host_1 | run5 | Done | 4283 | 30 (1%) | 22 (1%) | 0h 0m 40s |
| MapLogOptRegDup | host_1 | run6 | Done | 701 | 30 (1%) | 32 (1%) | 0h 0m 40s |
| MapExtraEffort2 | host_1 | run7 | Done | 784 | 30 (1%) | 32 (1%) | 0h 0m 40s |

## まとめ

この演習では、SmartXplorer で合成とインプリメンテーションを実行し、ビルトイン ストラテジを実行して結果を取得しました。

また、最良のストラテジを 5 つの異なるコスト テーブルを使用して実行し、タイミングを向上しました。

その後、あらかじめ定義されているストラテジと追加の反復を一度に実行する方法を説明しました。

最後に、SmartXplorer をインプリメンテーション フローのみで実行する方法を示しました。

XILINX®

第 5 章

# 演習 2 : カスタム ストラテジの作成

## 目標

7 つのビルトイン ストラテジを実行した結果、2 つのストラテジでほかの 5 つのストラテジよりよい結果が得られることがわかったので、今後の実行ではこれら 2 つのストラテジのみを実行し、ほかの 5 つはスキップします。

また、デザインで最適な結果を得られるストラテジを手動で見つけた場合、これらのストラテジを SmartXplorer で実行できると便利です。

どちらの場合でも、カスタム ストラテジ ファイルを作成し、SmartXplorer で使用できます。

この演習では、カスタム ストラテジを作成し、SmartXplorer で実行する方法を示します。

## 演習

### 手順 1 : ザイリンクス環境の設定

1. ターミナル ウィンドウを開き、ザイリンクス環境を設定します。
2. `cd` コマンドを使用して `lab2` ディレクトリに移動します。

### 手順 2 : 2 つの最良のストラテジの特定

演習 1 では、次のビルトイン ストラテジで良い結果が得られました。

- 合成 : XSTOnehotRedcon および XSTOptOnehotRedcon
- インプリメンテーション : MapGlobOptLogOptRegDup および MapRegDup

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| XSTOptReshRedcon_MapRunTime | host_1 | run1 | Done | 2118 | 30 (1%) | 29 (1%) | 0h 1m 5s |
| XSTOptOnehot_MapRunTime | host_1 | run2 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehot_MapRunTime | host_1 | run3 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 59s |
| XSTOptOnehotRedcon_MapRunTime | host_1 | run4 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 50s |
| XSTRegbalOptOnehot_MapRunTime | host_1 | run5 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehotRedcon_MapRunTime | host_1 | run6 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 49s |
| XSTRegbalOptOnehotReshRedcon_MapRunTime | host_1 | run7 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDup | host_1 | run8 | Done | 34 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptIOReg | host_1 | run9 | Done | 5784 | 48 (2%) | 22 (1%) | 0h 0m 45s |
| XSTOnehotRedcon_MapRegDup | host_1 | run10 | Done | 561 | 30 (1%) | 32 (1%) | 0h 0m 39s |
| XSTOnehotRedcon_MapExtraEffortIOReg | host_1 | run11 | Done | 4283 | 30 (1%) | 22 (1%) | 0h 0m 39s |
| XSTOnehotRedcon_MapLogOptRegDup | host_1 | run12 | Done | 701 | 30 (1%) | 32 (1%) | 0h 0m 39s |
| XSTOnehotRedcon_MapExtraEffort2 | host_1 | run13 | Done | 784 | 30 (1%) | 32 (1%) | 0h 0m 39s |

## 手順 3 : カスタム ストラテジ ファイルの作成

手順 2 で選択したストラテジに基づくカスタム ストラテジを作成する前に、これらのストラテジで使用されているオプションを確認します。

この情報は、lab1\smartxplorer_results ディレクトリにある smartxplorer.txt ファイルに保存されています。このファイルには、前回実行されたストラテジのオプションがすべて含まれます。次のファイルに、このファイルの内容が含まれています。

lab2 ディレクトリの lab1_smartxplorer.txt ファイル

```
------------------------------------------------------------------------
Strategy :XSTOnehotRedcon_MapRunTime
------------------------------------------------------------------------
Run index   : run6
Xst options :-fsm_encoding one-hot -reduce_control_sets auto
Map options :-ol high -w
Par options :-ol high

------------------------------------------------------------------------
Strategy :XSTRegbalOptOnehotReshRedcon_MapRunTime
------------------------------------------------------------------------
Run index   : run7
Xst options :-register_balancing yes -opt_level 2 -fsm_encoding one-hot
-resource_sharing no -reduce_control_sets auto

Map options :-ol high -w
Par options :-ol high

------------------------------------------------------------------------
Strategy :XSTOnehotRedcon_MapGlobOptLogOptRegDup
------------------------------------------------------------------------
Run index   : run8
Xst options :-fsm_encoding one-hot -reduce_control_sets auto
Map options :-ol high -xe n -global_opt speed -logic_opt on
-register_duplication on -w

Par options :-ol high -xe n
------------------------------------------------------------------------
 Strategy :XSTOnehotRedcon_MapRegDup
------------------------------------------------------------------------
Run index   : run10
Xst options :-fsm_encoding one-hot -reduce_control_sets auto
Map options :-ol high -xe n -register_duplication on -w
Par options :-ol high -xe n
```

この情報を使用して、ストラテジ ファイルを作成します。ストラテジの名前を次のように変更します。

- XSTOnehotRedcon を Syn1 に、XSTOptOnehotRedcon を Syn2 に変更
- MapGlobOptLogOptRegDup を Impl1 に、MapRegDup を Impl2 に変更

  lab2 ディレクトリにある my_strategy.txt ファイルを開きます。このファイルに Syn1、Syn2、Impl1、および Impl2 の定義が含まれています。

```
{
"spartan6":
(
{"name":"Syn1", "xst":"-fsm_encoding one-hot -reduce_control_sets auto"},
{"name":"Syn2", "xst":"-register_balancing yes -opt_level 2 -fsm_encoding one-hot
-resource_sharing no -reduce_control_sets auto"},
    ),

    "MAP-Par options":
    (
      {"name":"Impl1",
      "map":"-ol high -xe n -global_opt speed -logic_opt on -register_duplication
on -w",
       "par":"-ol high -xe n"},
      {"name":"Impl2",
       "map":"-ol high -xe n -register_duplication on -w",
       "par":"-ol high -xe n"},
    ),
  },
}
```

1. SmartXplorer コマンドで、**-sf** オプションを使用して my_strategy.txt ファイルを指定します。
2. ターミナル ウィンドウに次のコマンドを入力して SmartXplorer を起動します。

   ```
   smartxplorer -p xc6slx4-3-tqg144 -uc stopwatch.ucf
   -wd smartxplorer_results -sd ipcore_dir -sf my_strategy.txt
   stopwatch.xst
   ```

3. SmartXplorer で次の結果が生成されます (smartxplorer.html を参照)。

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| Syn1_Impl1 | host_1 | run1 | Done | 34 | 48 (2%) | 32 (1%) | 0h 0m 54s |
| Syn2_Impl1 | host_1 | run2 | Done | 34 | 48 (2%) | 32 (1%) | 0h 0m 54s |
| Syn2_Impl2 | host_1 | run3 | Done | 561 | 30 (1%) | 32 (1%) | 0h 0m 39s |

カスタム ストラテジを使用した SmartXplorer の実行は、あらかじめ定義されたストラテジを使用した SmartXplorer の実行と同様です。

- まず、すべてのカスタム合成ストラテジが実行され、カスタム インプリメンテーション ストラテジの最初のストラテジ (Impl1) がクイック インプリメンテーション ストラテジとして使用されます。
- すべての合成ストラテジの実行が完了すると、タイミング スコアとランタイムに基づいて最良のネットリストが選択され (この例では Syn2)、そのネットリストが残りのカスタム インプリメンテーション ストラテジ (この例では Impl2) を使用して実行されます。

# まとめ

この演習では、カスタム ストラテジ ファイルを作成し、それを使用してデザインを実行しました。

XILINX®

第 6 章

# 演習 3：複数のストラテジの同時実行

**重要** ：この演習は、次の場合にのみ実行可能です。

- 通常の Linux ネットワーク、LSF および SGE コンピューター ファームへのアクセスがある場合
- マルチコア プロセッサまたは複数のプロセッサを持つマシンを使用している場合

## 目標

この演習では、通常の Linux ネットワーク、LSF および SGE コンピューター ファーム、マルチコア プロセッサまたは複数のプロセッサを持つマシンでザイリンクス環境を設定し、SmartXplorer を使用する方法を説明します。

これらのネットワークで SmartXplorer を使用する場合の考慮事項を示します。

## 演習

### 手順 1：ザイリンクス環境と結果の保存

SmartXplorer を起動する前に、次の 2 つの事項を理解する必要があります。

- 各マシンでのザイリンクス ソフトウェアの環境設定方法
- SmartXplorer の結果の保存場所

1. ザイリンクス環境の設定

   デザイン ストラテジの実行に 3 つの Linux マシン (L1、L2、および L3) を使用する通常の Linux ネットワークの場合を考えてみます。SmartXplorer の起動には L1 を使用します。

   L2 (L3) でジョブを起動する前に、SmartXplorer により自動的に L2 (L3) マシンで $XILINX 環境変数が L1 の $XILINX と同じ値に設定されます。これは、次のことを意味します。

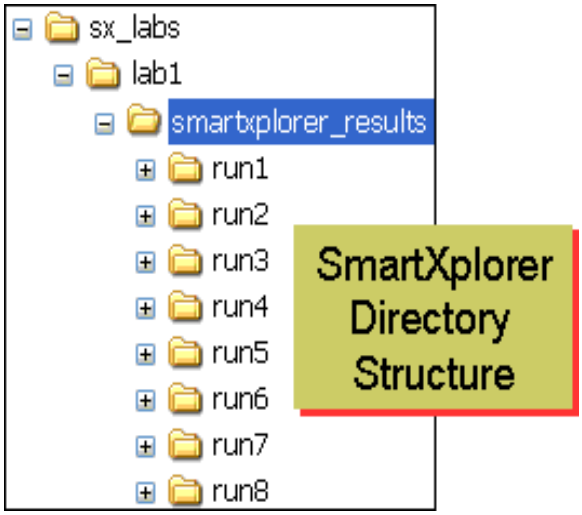
   - ザイリンクス ソフトウェアがネットワーク上にインストールされている場合、L2 (L3) でそのソフトウェアにアクセスする必要があり、また L1 に定義されたネットワーク パスがすべてのマシンで有効となるよう同じネットワーク割り当てポイントを使用する必要があります。
   - ザイリンクス ソフトウェアが L1 のローカル ディスクにインストールされている場合、L2 (L3) のローカル ディスクの L1 と同じパスに同じバージョンのザイリンクス ソフトウェアがインストールされていることが必要です。

環境変数がこのように設定されるのは、各デザイン ストラテジが同じ条件下で実行されるようにするためです。$XILINX に加え、L1 で設定されているすべてのザイリンクス環境変数 (接頭辞「XIL_」で始まるもの) が検出され、L2 および L3 に適用されます。

LSF および SGE コンピューター ファームを使用する場合でも、ザイリンクス ソフトウェアを実行する各マシンで、SmartXplorer を実行したマシンと同じザイリンクス環境変数が設定されている必要があります。

2. 結果の保存場所

   演習 1 で説明したように、SmartXplorer の結果はすべて `smartxplorer_results` という別ディレクトリに保存されます。




複数のマシンを使用する場合でも同じで、結果はすべて同じディスク エリアに保存されます。そのため、すべてのマシンにこのディスク エリアに対する読み取り/書き込み権限が必要です。

## 手順 2 : 実行前のチェックリスト

1. 使用するマシンのリストを作成します。
2. 各マシンで同じザイリンクス環境が設定されていることを確認します。
3. すべてのマシンに SmartXplorer の結果が保存されるディレクトリへの読み出し/書き込み権限があることを確認します。

## 手順 3 : ザイリンクス環境の設定

1. ターミナル ウィンドウを開き、ザイリンクス環境を設定します。
2. **`cd`** コマンドを使用して `lab3` ディレクトリに移動します。

## 手順 4：通常の Linux ネットワークでの SmartXplorer の実行

2 つの Linux マシンhost_1 および host_2 を使用して、7 つのビルトイン ストラテジを実行します。host_1 上では 2 つのジョブ、host_2 では 1 つのジョブを同時に実行できるとします。SmartXplorer の実行には、host_1 を使用します。

1. lab3 ディレクトリにある my_strategy.txt ファイルをテキスト エディターで開きます。このファイルには、デザイン ストラテジを実行するのに使用するマシンがリストされています。このファイルには、各マシンの名前を個別の行に記述します。

```
host_1
host_1
host_2
```

   同じマシンで 2 つのジョブを同時に実行できるようにするには、上記の例の host_1 のように、そのマシンを 2 回リストする必要があります。

   **メモ**：my_hostlist.txt ファイルで host_1 および host_2 を実際に使用するマシンの名前に置き換え、ファイルを保存します。

2. **-l** オプションを使用して my_hostlist.txt ファイルを指定します。
3. ターミナル ウィンドウに次のコマンドを入力して SmartXplorer を起動します。

```
smartxplorer -p xc6slx4-3-tqg144 -uc stopwatch.ucf -wd
smartxplorer_results -sd ipcore_dir -l my_hostlist.txt stopwatch.xst
```

4. 次のような表が表示されます。host_1 で 13 個のうち 9 個のストラテジが実行されました。

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| XSTOptReshRedcon_MapRunTime | host_1 | run1 | Done | 2118 | 30 (1%) | 29 (1%) | 0h 1m 19s |
| XSTOptOnehot_MapRunTime | host_1 | run2 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTOnehot_MapRunTime | host_2 | run3 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 1m 35s |
| XSTOptOnehotRedcon_MapRunTime | host_1 | run4 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 54s |
| XSTRegbalOptOnehot_MapRunTime | host_2 | run5 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 1m 29s |
| XSTOnehotRedcon_MapRunTime | host_1 | run6 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 45s |
| XSTRegbalOptOnehotReshRedcon_MapRunTime | host_1 | run7 | Done | 1895 | 30 (1%) | 32 (1%) | 0h 0m 55s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDup | host_1 | run8 | Done | 34 | 48 (2%) | 32 (1%) | 0h 0m 49s |
| XSTOnehotRedcon_MapGlobOptIOReg | host_1 | run9 | Done | 5784 | 48 (2%) | 22 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapRegDup | host_2 | run10 | Done | 561 | 30 (1%) | 32 (1%) | 0h 1m 33s |
| XSTOnehotRedcon_MapExtraEffortIOReg | host_1 | run11 | Done | 4283 | 30 (1%) | 22 (1%) | 0h 0m 40s |
| XSTOnehotRedcon_MapLogOptRegDup | host_1 | run12 | Done | 701 | 30 (1%) | 32 (1%) | 0h 0m 40s |
| XSTOnehotRedcon_MapExtraEffort2 | host_2 | run13 | Done | 784 | 30 (1%) | 32 (1%) | 0h 1m 40s |

**メモ**：1 つのストラテジが完了し、タイミングが満たされると、現在実行中のストラテジの実行がすべて停止します。これは、複数のストラテジが同時に実行されており、**-ra** がオプションが使用されていない場合の動作です。

| Strategy | Host | Output | Status | Timing Score | Luts | Slice Registers | Total RunTime |
|---|---|---|---|---|---|---|---|
| XSTOnehotRedcon_MapExtraEffort2 | host_2 | run13 | Done | 784 | 30 (1%) | 32 (1%) | 0h 1m 40s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDupCT2 | host_1 | run14 | Done | 129 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDupCT3 | host_1 | run15 | Done | 0 | 48 (2%) | 32 (1%) | 0h 0m 44s |
| XSTOnehotRedcon_MapGlobOptLogOptRegDupCT4 | host_2 | run16 | Stopped | None | 48 (2%) | 32 (1%) | 0h 1m 5s |

上の図で、赤色の長方形で囲まれている 2 つのストラテジのステータスを見てみます。
XSTOnehotRedcon_MapGlobOptLogOptRegDupCT3 の後に
XSTOnehotRedcon_MapGlobOptLogOptRegDupCT4 が開始していますが、
XSTOnehotRedcon_MapGlobOptLogOptRegDupCT4 が完了する前に
XSTOnehotRedcon_MapGlobOptLogOptRegDupCT3 でタイミングが満たされたため、
XSTOnehotRedcon_MapGlobOptLogOptRegDupCT4 の実行が停止され、[Status] 列に [Stopped] と示されています。

## 手順 5 : LSF または SGE での SmartXplorer の実行

LSF および SGE コンピューター ファームを使用するのは、通常の Linux ネットワークを使用するのとほとんど同じで、ホスト リスト ファイルを作成して [Use host list file] で指定します。

主な違いは、ホスト リスト ファイルでのホスト マシンの定義方法です。LSF および SGE コンピューター ファームのフォーマットは、次のとおりです。

| | |
|---|---|
| LSF | :LSF {"queue_name":"MYQUEUE", "max_concurrent_runs":N, "bsub_options":"additional_options"} |
| SGE | :SGE {"queue_name":"MYQUEUE", "max_concurrent_runs":N, "qsub_options":"additional_options"} |

説明 :

- queue_name : キューの名前を指定します。「MYQUEUE」を LSF または SGE キューの名前に置き換えてください。
- max_concurrent_runs : 同時に実行可能なジョブの最大数を指定します。N に正の整数を指定してください。
- bsub_options : 追加の LSF オプションを指定します。「additional_options」を LSF オプションに置き換えてください。オプションを使用しない場合は、“” を使用します。
- qsub_options : 追加の SGE オプションを指定します。「additional_options」を SGE オプションに置き換えてください。オプションを使用しない場合は、“” を使用します。

例 :

キューの名前が lin64_q、並行して実行可能なジョブの最大数が 6、指定する LSF または SGE オプションがない場合は、ホスト リスト ファイルに次のように記述します。

| | |
|---|---|
| LSF | :LSF {"queue_name":"lin64_q", "max_concurrent_runs":6, "bsub_options":""} |
| SGE | :SGE {"queue_name":"lin64_q", "max_concurrent_runs":6, "qsub_options":""} |

# まとめ

この演習では、複数の Linux マシンで SmartXplorer を使用して複数のデザイン ストラテジを同時に実行する方法を説明しました。

これらのネットワークで SmartXplorer を使用する場合の考慮事項を示しました。

XILINX®

# 付録 A

# カスタム ファイル

## 目標

この付録では、次のものの例を示します。

- カスタム ストラテジ ファイル
- ホスト リスト ファイル (Linux)
- さまざまなタスクでの SmartXplorer の設定 (合成およびインプリメンテーション フロー )
- さまざまなタスクでの SmartXplorer の設定 (インプリメンテーション フローのみ)

## カスタム ストラテジ ファイル

次に、カスタム ストラテジ ファイルの例を示します。Spartan®-6 デバイスの 2 つのストラテジと Virtex®-6 デバイスの 2 つのストラテジが含まれています。

```
{
  "spartan6":
  {
    "XST Options":
    (
     {"name":"Syn1", "xst":"-fsm_encoding one-hot -reduce_control_sets
auto"},
      {"name":"Syn2", "xst":"-register_balancing yes -opt_level 2 -
fsm_encoding one-hot -resource_sharing no -reduce_control_sets auto"},
    ),

    "MAP-Par options":
    (
      {"name":"Impl1",
       "map":"-ol high -xe n -global_opt speed -logic_opt on -
register_duplication on -w",
       "par":"-ol high -xe n"},
      {"name":"Impl2",
       "map":"-ol high -xe n -register_duplication on -w",
       "par":"-ol high -xe n"},
    ),
  },

  "virtex6":
  {
    "XST Options":
    (
      {"name":"Syn3", "xst":"-fsm_encoding one-hot -opt_level 2"},
```

```
      {"name":"Syn4", "xst":"-register_balancing yes -fsm_encoding one-
hot -resource_sharing no -reduce_control_sets auto"},
    ),

    "MAP-Par options":
    (
      {"name":"Impl3",
       "map":"-ol high -xe n -global_opt speed -logic_opt on -w",
       "par":"-ol high -xe n"},
      {"name":"Impl4",
       "map":"-ol high -xe n -w",
       "par":"-ol high -xe n"},
    ),
  },
}
```

# ホスト リスト ファイル (Linux)

## 通常の Linux ネットワーク

次に、通常の Linux ネットワーク用のホスト リスト ファイルの例を示します。

```
lin_machine_1
lin_machine_1
lin_machine_2
lin_machine_3
```

この例では、SmartXplorer で 4 つのストラテジが同時に実行されます。3 つの Linux マシンが使用され、lin_machine_1 では 2 つのストラテジが同時に実行されます。

## LSF コンピューター ファーム

次に、LSF コンピューター ファーム用のホスト リスト ファイルの例を示します。この例では、キュー名は lin64_q で、同時に実行可能なジョブの最大数は 6 です。

```
:LSF {"queue_name":"lin64_q", "max_concurrent_runs":6, "bsub_options":""}
```

## SGE コンピューター ファーム

次に、SGE コンピューター ファーム用のホスト リスト ファイルの例を示します。この例では、キュー名は lin64_q で、同時に実行可能なジョブの最大数は 6 です。

```
:SGE {"queue_name":"lin64_q", "max_concurrent_runs":6, "qsub_options":""}
```

# さまざまなタスクでの SmartXplorer の設定 (合成およびインプリメンテーション フロー )

## タスク 1 : すべてのビルトイン ストラテジを実行

次の例は、すべてのビルトインの合成およびインプリメンテーション ストラテジを実行します。**-m** はデフォルト値 13 を使用するので記述されていません。ホスト リスト ファイル (**-l** オプション) で指定されているマシン上でストラテジが実行されます。タイミングが満たされると、実行は終了します。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file>
<design>.xst
```

## タスク 2 : 最初の 3 つのビルトイン ストラテジを実行

次の例は、ビルトイン ストラテジのうち最初の 3 つの合成ストラテジとクイック インプリメンテーション ストラテジを実行します (**-m 3** オプション)。ホスト リスト ファイルで指定されているマシン上 (**-l** オプション) でストラテジが実行されます。タイミングが満たされると、実行は終了します。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file>
-m 3 <design>.xst
```

## タスク 3 : ビルトイン ストラテジすべてを実行し、異なるコスト テーブルを使用して追加で 5 回実行

**段階 1** : 13 個のビルトイン ストラテジすべてを実行します。

**段階 2** : 段階 1 から最良のストラテジを選択し、そのストラテジを異なるコスト テーブルを使用して追加で 5 回実行します (**-m 18** オプション、13 + 5 = 18)。

ホスト リスト ファイルで指定されているマシン上 (- l オプション) でストラテジが実行されます。

タイミングが満たされると、実行は終了します。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file>
-m 18 <design>.xst
```

## タスク 4 : カスタム ストラテジすべてを実行し、異なるコスト テーブルを使用して追加で 3 回実行

この例では、カスタム ストラテジ ファイルに 2 つの合成ストラテジと 2 つのインプリメンテーション ストラテジが含まれているとします。

**段階 1** : 2 つの合成ストラテジを最初のインプリメンテーション ストラテジと共に実行します。

**段階 2** : 段階 1 から最良の合成ストラテジを選択し、2 番目のインプリメンテーション ストラテジと共に実行します。

**段階 3** : 段階 1 および 2 から最良のストラテジの組み合わせを選択し、その組み合わせを異なるコスト テーブルを使用して追加で 3 回実行します (**-m 6** オプション)。**-m** の値は、X を合成ストラテジ数、Y をインプリメンテーション ストラテジ数、Z を追加で実行する反復回数とすると、X + Y + Z - 1 で求められます。

ホスト リスト ファイルで指定されているマシン上 (- l オプション) でストラテジが実行されます。

途中でタイミングが満たされた場合でも、実行が 6 回反復されます (**-ra** オプション)。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file>
-m 6 -sf <strategy_file> -ra <design>.xst
```

または、次のように -vp オプションを使用して、異なるコスト テーブルを使用して実行する反復回数を指定できます。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file>
-vp 3 -sf <strategy_file> -ra <design>.xst
```

# さまざまなタスクでの SmartXplorer の設定 (インプリメンテーション フローのみ)

## タスク 1 : すべてのビルトイン ストラテジを実行

次の例は、すべてのビルトイン インプリメンテーション ストラテジを実行します。**-m** はデフォルト値 7 を使用するので記述されていません。ホスト リスト ファイルで指定されているマシン上 (**-l** オプション) でストラテジが実行されます。タイミングが満たされると、実行は終了します。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file> <design>.ngc
```

## タスク 2 : 最初の 3 つのビルトイン ストラテジを実行

次の例は、ビルトイン ストラテジのうち最初の 3 つのインプリメンテーション ストラテジを実行します (**-m 3** オプション)。ホスト リスト ファイルで指定されているマシン上 (**-l** オプション) でストラテジが実行されます。タイミングが満たされると、実行は終了します。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file>
-m 3 <design>.ngc
```

## タスク 3 : ビルトイン ストラテジすべてを実行し、異なるコスト テーブルを使用して追加で 5 回実行

**段階 1** : 7 個のビルトイン インプリメンテーション ストラテジすべてを実行します。

**段階 2** : 段階 1 から最良のストラテジを選択し、そのストラテジを異なるコスト テーブルを使用して追加で 5 回実行します (**-m 12** オプション、7 + 5 = 12)。

ホスト リスト ファイルで指定されているマシン上 (- l オプション) でストラテジが実行されます。

タイミングが満たされると、実行は終了します。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file>
-m 12 <design>.ngc
```

## タスク 4 : カスタム ストラテジすべてを実行し、異なるコスト テーブルを使用して追加で 3 回実行

この例では、カスタム ストラテジ ファイルに 2 つのインプリメンテーション ストラテジが含まれているとします。

**段階 1** : カスタム ストラテジ ファイルで指定されている 2 つのストラテジを実行します (**-sf** オプション)。

**段階 2** : 段階 1 から最良のストラテジを選択し、そのストラテジを異なるコスト テーブルを使用して追加で 3 回実行します (**-m 5** オプション、2 + 3 = 5)。

ホスト リスト ファイルで指定されているマシン上 (**-l** オプション) でストラテジが実行されます。

タイミングが満たされた場合でも、実行が 5 回反復されます (**-ra** オプション)。

```
smartxplorer -p xc6slx4-3-tqg144 -uc <file>.ucf -l <host_list_file>
-m 5 -sf <strategy_file> -ra <design>.ngc
```

# 付録 B

# XST のスクリプト ファイルの作成

ザイリンクス ツールをコマンド ライン モードで使用している場合は、コマンド ライン モードで XST を実行するのに使用するのと同じ *design*.xst スクリプト ファイルを使用する必要があります。

ISE® Project Navigator がデフォルトのデザイン環境である場合は、プロジェクト ディレクトリに *design*.xst が含まれています。このファイルは、Project Navigator から XST を実行すると自動的に生成されます。これを SmartXplorer を起動する際に入力ファイルとして使用する必要があります。

design.xst から、次のようなファイルおよびディレクトリが参照されることがあります。

- *design*.prj : XST プロジェクト ファイル (必須)
- *design*.xcf : XST 制約ファイル
- *design*.lso ： ライブラリ検索順ファイル
- その他

そのため、SmartXplorer をプロジェクト ディレクトリから起動することをお勧めします。