階層デザイン手法ガイド

UG748 (v13.3) 2011 年 10 月 19 日

本資料は英語版 (v 13.3) 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。

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 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

2011 年 3 月 1 日	13.1	• 「チーム デザイン フロー」の章を追加
		• 次に関する新しい情報を追加
		  • ブラック ボックスのサポート
		  • ImportTag
		  • メモリ量の削減方法

2011 年 7 月 6 日	13.2	• 「パーティションのステート」セクションにパーティションのステートを **auto** に設定した場合の情報を追加 • 「プロセッサ システムのパーティション」セクションを追加 • 「インポートするパーティションの属性値について」セクションを追加 • 一部の機能が Virtex®-6、Spartan®-6、および 7 シリーズ FPGA デバイスでのみサポートされることを記述 • 「ImportTag 属性の設定」セクションに Tcl コンソールからの ImportTag 属性の設定に関する情報など、新しい情報を追加 • 「Preservation 属性の管理」セクションで、コマンドを [Implementation Settings] から [Specify Partitions] に修正
2011 年 10 月 19 日	13.3	• 「PXML ファイルのエクスポート」セクションの最初の手順をわかりやすく記述 • 「パーティションのエクスポート」セクションに PREV ファイルの情報を追加 • 「最上位」と「トップ パーティション」を明確に区別 • 「チーム メンバー ブロックを特定の領域にフロアプラン」セクションに情報を追加

目次

改訂履歴 . . . 3

第 1 章 : パーティション

PXML ファイル . . . 7
パーティションを使用する状況の判断 . . . 7
パーティション使用の利点と欠点 . . . 8
パーティションのステート . . . 8
再インプリメントが必要なパーティションの変更 . . . 9
再インプリメントが不要なパーティションの変更 . . . 10
パーティションの保持レベル . . . 10
インポート ディレクトリ . . . 11
異なる階層へのインポート . . . 11
大型デザインのメモリ使用量の管理 . . . 12
ブラック ボックスの使用法 . . . 13
パーティションのコンテキスト ルール . . . 16

第 2 章 : 設計に関する考慮事項

最適化の制限 . . . 19
BoundaryOpt 属性を使用した IP コアの最適化 . . . 22
デザインの構造 . . . 24
階層デザイン フローの有効活用 . . . 25
配線情報の保持に関する制限 . . . 28
パーティションのフロアプラン . . . 29
プロセッサ システムのパーティション . . . 30

第 3 章 : 合成パーティションフロー

インクリメンタル合成パーティション フロー . . . 31
ボトムアップ合成パーティション フロー . . . 32
合成ツール . . . 33

第 4 章 : コマンド ラインでのパーティション フロー

ザイリンクス インプリメンテーション ツール . . . 39
PXML ファイルの作成 . . . 39
インプリメンテーションの実行 . . . 42
パーティションのエクスポート . . . 43
パーティションのステートを import に変更 . . . 44
デザインでの反復作業 . . . 45
パーティションを含むデザインでの SmartXplorer の使用 . . . 45
パーティションの削除と復元 . . . 46

第 5 章 : PlanAhead でのパーティション フロー

PlanAhead でのパーティション フローについて . . . 49
新規プロジェクトの作成 . . . 50
パーティションの設定 . . . 50
PXML ファイルのインポート . . . 51
PXML ファイルのエクスポート . . . 51

BoundaryOpt 属性の設定 52
ImportTag 属性の設定 52
パーティションのフロアプラン 53
パーティション デザインの合成 53
パーティション デザインのインプリメント 53
パーティションのプロモート 55
パーティション属性の管理 55
デザイン実行の管理 57

第 6 章 : デザイン保持フロー
インプリメンテーションのランタイム 59
コマンド ラインでのデザイン保持フロー 60
PlanAhead ソフトウェアでのデザイン保持フロー 61
ネットリスト デザイン保持フロー 62

第 7 章 : チーム デザイン フロー
チーム デザイン フローのチーム 63
チーム デザイン フローの概要 64
チーム デザイン フローのセットアップ 65
チーム メンバーの役割り 70
チーム リーダーの役割り 73
すべてのパーティション デザインの設計に関する推奨事項 77
チーム デザイン フロー特定の設計に関する推奨事項 77
コマンド ライン フロー 80
PlanAhead ソフトウェア フロー 83
インターフェイス タイミング 86

第 8 章 : パーティションのデバッグ
インプリメンテーション エラー 89
インポートするパーティションの属性値について 93
BitGen の DRC エラー 94
ChipScope のサポート 94

付録 A : その他のリソース
ザイリンクス リソース 81
ISE 資料 81
PlanAhead 資料 82

XILINX®

第 1 章

パーティション

階層デザイン フローでは、デザインを「パーティション」と呼ばれる小型のブロックに分割します。パーティションには、次のような特徴があります。

- ザイリンクス ソフトウェアの階層デザイン フローの基本ブロックです。
- 階層の境界を定義します。
- 複雑なデザインを小型の作業しやすいブロックに分割します。
- 階層モジュール インスタンスの周囲に境界を作成し、デザインのほかの部分から分離します。
- インプリメントおよびエクスポートしたパーティションは、単純なコピー / 貼り付け操作によりデザインに挿入でき、配置配線結果が保持されます。

PXML ファイル

パーティションは、PXML ファイルを使用して定義および制御します。

PXML ファイルとは、次のようなファイルです。

- `xpartition.pxml` という名前が付いています。
- ツールを実行すると読み込まれます。
- 次の方法で作成できます。
  - PXML テンプレートを使用するかまたは使用せずに、手動で作成
  - PlanAhead™ のグラフィカル ユーザー インターフェイス (GUI) を使用して作成

PXML ファイルの作成については、第 4 章「コマンド ラインでのパーティション フロー」を参照してください。

パーティションを使用する状況の判断

パーティションは、パーティションが必要なモジュールにのみ使用してください。パーティションを過度に使用すると、ランタイムが増加し、パフォーマンスが低下することがあります。

次のようなモジュールでは、フラット最適化を使用した方が良い結果が得られます。

- 独立したファンクション ブロックではなく、個別の階層を持たない
- ほかのブロックとのグローバル最適化が有効

パーティションを効果的に使用するための考慮事項は、第 2 章「設計に関する考慮事項」を参照してください。

パーティションを設定するのに適しているのは、次のものです。

- 次のようなファンクション ブロック
  - DSP モジュール
  - EDK システム
- 高パフォーマンス コア
- デバイスに一緒に配置する必要のあるロジックを含むインスタンス
- デザイン ガイドラインに従ったモジュール

パーティション使用の利点と欠点

階層フローには、利点と欠点があります。パーティションを使用すると、パーティションの境界を越えた最適化は実行されません。

その点を考慮してデザインを設計しないと、パーティションを使用することがタイミング、リソース使用量、ランタイムに大きく影響を与える可能性があります。注意深く設計しても、最適化および配置に関するほかの制限もあるので、リソース使用量が増加し、タイミングが悪化する場合があります。適切に設計されたデザインではこれらの影響は最小限に抑えられますが、これらの制限を念頭に置いておくことは重要です。

パーティションの最適化への影響およびこれらの影響を最小限に抑える設計方法については、第 2 章「設計に関する考慮事項」を参照してください。

パーティションのステート

パーティションは、そのステートによってインプリメントまたはインポートできます。

パーティションのステートを import に設定

パーティションを初めて ISE® Design Suite インプリメンテーション ツールで実行する場合は、パーティションのステートを implement に設定します。インプリメンテーション ツールには、次のものがあります。

- NGDBuild
- MAP
- PAR

パーティションのエクスポート

インプリメンテーションが完了したら、パーティションをエクスポートし、今後の実行で結果をインポートできます。

エクスポートされた結果は、内部ロジックおよびパーティション インターフェイスが変更されていない場合にのみ使用可能です。

パーティションのステートを auto に設定

パーティションのステートを **auto** に設定できます。

auto に設定すると、NGDBuild で次の処理が実行されます。

- 論理的に変更されていないパーティションはインポートされます。
- 論理的に変更されているパーティションはインプリメントされます。

パーティションのステートを **auto** に設定する場合、次のような条件があります。

- コマンド ライン フローからのみサポートされます。
- **ImportLocation** 属性を設定する必要があります。

  ImportLocation 属性で指定するディレクトリは最初の実行では空でもかまいませんが、インポートするパーティションがある場合は、インプリメントされたパーティションがエクスポートされているディレクトリに設定する必要があります。
- **auto** ステートは、NGDBuild で処理されるデザインへの論理的な変更のみに使用できます。
  - LOC や AREA_GROUP 制約の変更などの物理的な変更は、マップの段階まで処理されません。
  - その場合は、ステートを **implement** に手動で変更する必要があります。

再インプリメントが必要なパーティションの変更

パーティション モジュールに変更を加えた場合、そのパーティションの配置配線が最新ではなくなり、変更したモジュールの再インプリメントが必要になります。変更していないパーティションは、前回の実行結果をインポートできます。

パーティションに次の変更を加えた場合に、再インプリメントが必要になります。

- ハードウェア記述言語 (HDL) コードの変更
- パーティションに関連したネットリストが変更される変更
- パーティションに関連する次のような物理制約の変更 (**state=auto** では処理されない)
  - AREA_GROUP
  - LOC
- 次のようなターゲット アーキテクチャの変更
  - デバイス
  - パッケージ
  - スピード グレード
- パーティションに接続された ChipScope™ Analyzer コアの追加および接続の変更
- パーティション インターフェイスのコンテキストの変更

  この章の「パーティションのコンテキスト ルール」を参照してください。

エクスポートされたパーティションのアップデートが必要な場合は、パーティションのステートを正しく管理するため、PXML ファイルで **state** 属性を変更します。変更しないと、インプリメンテーション エラーが発生します。

再インプリメントが不要なパーティションの変更

次のようなパーティションの変更では、再インプリメントは必要ありません。

- ロジックの物理ロケーションに影響しない制約の変更 (例 : TIMESPEC)
- インプリメンテーション オプションの変更 (例 : **`par -xe`**)

パーティションの保持レベル

パーティションを使用すると、以前の結果をインポートすることにより実行結果を保持できます。

パーティションをインポートする際、次を実行できます。

- 保持レベルを指定します。デフォルトでは、配置配線が 100% 保持されます。
- デフォルトを次を保持するよう変更します。
  - 配置結果

    配線は変更される場合があります。
  - 合成結果

    配置および配線が変更される場合があります。
- 保持レベルに基づいて、次の目的で多少の変更を加えます。
  - タイミングの向上
  - 配置または配線の競合の解決

  インポートするパーティションに対しては、保持レベルにかかわらず、まずすべての配置配線情報がコピーされます。

保持レベルは、インプリメンテーション ツールでインポートされた配置配線をどれだけ変更できるかを指定するものです。保持レベルを緩和するとデバイス リソースが解放され、インプリメンテーション ツールでほかのパーティションをより柔軟に配置配線できるようになります。

保持レベルは、次のように適用されます。

- パーティションごとに設定できます。
- インポートするパーティションのみに適用されます。

タイミング クリティカルなパーティション モジュールでタイミングが満たされ、変更する予定がない場合は (IP コアなど)、配線までを保持するのが適切です。

パーティションがタイミング クリティカルでなく、タイミングが満たされていない場合は、保持レベルを緩和してツールでソリューションがより柔軟に見つけられるようにします。

パーティションを使用する目的が検証時間の短縮である場合は、常に保持レベルを配線に設定してください。次の目的で保持レベルを変更する必要がある場合は、パーティションを再検証してください。

- タイミングを満たす。
- デザインのほかの部分の配線を完了する。

パーティションをフロアプランすると、保持レベルを緩和する必要がなくなることがあります。

インポート ディレクトリ

パーティションをインポートする際は、エクスポートした結果の場所を指定する必要があります。

インプリメントされたデザインをエクスポートすると、デザインに含まれるすべてのパーティションがエクスポートされます。

チーム デザインや順次構築するフローでは、複数の場所または複数のエクスポート デザインからパーティションをインポートできます。

異なる階層へのインポート

パーティションは、インプリメントした元の階層とは異なる階層にインポートできます。

異なる階層レベルにパーティションをインポートするには、PXML ファイルで **ImportTag** 属性を使用します。この属性は、次のような場合に使用できます。

- 同じデザインのバリエーション間で最上位ブロックの名前が変更されている (Top、Top2 など)
- パーティションが次のような場合
  - その他のロジックは最小限でインプリメントされている (クロックおよび I/O を含む最上位)。
  - 階層が変更されるデザイン全体に追加されている。
- パーティションがまったく異なる階層にインポートされる。パーティションの入力および出力への接続は、一定している必要があります。

ImportTag 属性には、次のような特徴があります。

- 次のツールでサポートされます。
  - XST (Xilinx Synthesis Technology) 合成ツール
  - ISE Design Suite インプリメンテーション ツール
- 次の方法で定義できます。
  - 次で手動で
    - PXML ファイル
    - コマンド ライン フロー
  - PlanAhead ソフトウェアで属性を使用

PlanAhead ソフトウェアで **ImportTag** 属性を設定する方法については、第 5 章「PlanAhead でのパーティション フロー」を参照してください。

ImportTag 属性には、値としてインポートする元のパーティションを指定します。新しい階層は、パーティション名で定義されます。

次のように定義され、インプリメントされた元のパーティションがあるとします。

```
Partition Name="/iptop/ip"
```

このパーティションをエクスポートし、新しいデザインで次の名前で使用するとします。

```
Partition Name = "/top/x/y"
```

"/iptop/ip" という名前のパーティションを "/top/x/y" にインポートするには、**ImportTag** 属性を次のように設定します。

```
Partition Name="/iptop/ip"
```

大型デザインのメモリ使用量の管理

パーティションを使用すると、通常合計ピーク メモリ使用量が増加します。ほとんどの場合、この増加はたいした量ではありませんが、スライス数が 100,000 以上の大型デザインでは、増加量が大きくなる可能性があります。この増加は、パーティションのインポート元であるネットリスト制約定義 (NCD) ファイルのサイズが原因です。

複数のパーティションを含むデザインでは、次の方法を使用するとメモリ使用量の増加を抑えることができます。

- パーティションをブラック ボックスとして定義する
- 最小限のロジックを使用してパーティションを定義する

パーティションをブラック ボックスとして定義する

合成およびインプリメンテーションで、パーティションをブラック ボックスとして定義できます。

パーティションをブラック ボックスとして定義した場合の利点

1 つのパーティションにロジックが含まれており、その他のパーティションをブラック ボックスとして定義した場合にデザインをインプリメントすると、次のようになります。

- 生成される NCD ファイルのサイズが小さくなります。
- デザインに含まれるロジック数が少なくなります。
- 通常、ランタイムが短縮されます。
- インプリメンテーション ツールでデザインの一部分のみが処理されるので、パーティション内のタイミングが向上する場合があります。

ブラック ボックスとして定義する部分を増やすと、次のようになります。

- 生成される NCD ファイルのサイズが小さくなります。
- インポート時のメモリ使用量をより削減できます。

このプロセスを繰り返してすべてのパーティションをインプリメントし、それぞれの場所にエクスポートして、最終的な実行でそれらの場所から各パーティションをインポートします。

この方法は、第 7 章「チーム デザイン フロー」で説明されているフローと似ています。

パーティションをブラック ボックスとして定義した場合の欠点

パーティションをブラック ボックスとして定義すると、最終的にデザインをアセンブルしたときに、パーティションのインターフェイスのタイミングが満たされないことがあります。

1 つのパーティションをインプリメントするときにほかのパーティションはブラック ボックスとして定義されているので、欠けているロジックに接続されているタイミング パスに既存の制約が適用されません。ブラック ボックス モジュールへの接続は存在しますが、LUT1 としてインプリメントされるプロキシ ロジックに接続されます。

プロキシ ロジックに入出力するパスのタイミング制約は、次を使用して作成できます。

- FROM:TO 制約
- **TPSYNC** または **PPS**

ブラック ボックス モジュールのプロキシ ロジックに対する制約例は、「ブラック ボックスの使用法」を参照してください。

最小限のロジックを使用してパーティションを定義する

プロキシ ロジックとブラック ボックス モジュールを使用する方法のほかに、最小限のロジックを使用してパーティションを定義する方法もあります。

必要なロジックは、既存の制約を使用して現実的なタイミング結果を得るために必要なインターフェイス接続です。入力および出力がレジスタを介するようにすると、通常この要件は満たされます。

インプリメントされるパーティション以外のパーティションには最小限のロジックしか含まれていないので、次のようになります。

- 生成される NCD ファイルのサイズが完全なデザインよりかなり小さくなります。
- メモリ使用量が減少します。

インターフェイスのタイミングは次のようになります。

- より正確
- 最終デザインをアセンブルしたときに QoR が向上

インターフェイス ロジックを供給すると、次のようになります。

- 追加の設計時間が必要になります。
- プロキシ ロジックを使用する方法よりも適切です。
- ほかのパーティションへのインターフェイス タイミングが正確なものになるので、アセンブリ時の結果を予測しやすくなります。

ブラック ボックスの使用法

ISE Design Suite では、合成からインプリメンテーションまででブラック ボックス パーティションがサポートされます。

XST インクリメンタル フローを使用する場合は、次のようになります。

- PXML ファイルを使用してパーティションを定義します。
- パーティションをブラック ボックス モジュールに追加できます。

ポートの幅と方向を定義するモジュールまたはコンポーネント宣言のみが必要です。

ブラック ボックスがサポートされる状況

ブラック ボックス インスタンスが PXML ファイルでパーティションとして定義されていれば、ブラック ボックスはインプリメンテーションでサポートされます。これは、次の場合に有益です。

- チーム デザイン

  チーム メンバーのブロックが定義されておらず、使用できない場合

  詳細は、第 7 章「チーム デザイン フロー」を参照してください。
- メモリ量削減手法

  複数の小型の NCD ファイルを使用して個々のパーティションをエクスポートする場合

  詳細は、この章の「大型デザインのメモリ使用量の管理」を参照してください。

- 順次構築手法

  パーティションを 1 つずつ追加することで、インプリメンテーション ツールでデザインの一部分のみが集中して処理されます。これは、チーム デザイン フローを順次処理する手法と考えることができます。

  詳細は、第 7 章「チーム デザイン フロー」を参照してください。

警告メッセージ

ブラック ボックス パーティションを含むデザインをインプリメントすると、NGDBuild で次のような警告メッセージが表示されます。

```
WARNING:NgdBuild:1419 - Could not resolve Partition block 'top/u0'.
This might be legal if you are implementing the Partition as a
blackbox.If it is not a blackbox, you should check that the Partition's
module file, 'u0.ngc' or 'u0.edf' (or another valid EDIF extension,
exists and is properly set up in a search path (see the -sd option.
```

パーティション インプリメンテーションのステータス

NGDBuild レポート (.bld) の「Partition Implementation Status」(パーティション インプリメンテーション ステータス) セクションに、ブラック ボックス モジュールに関する情報も含まれます。

```
Partition Implementation Status
-------------------------------
 Preserved Partitions:

 Implemented Partitions:

 Partition "/eth_aggregator":
Attribute STATE set to IMPLEMENT.

 Partition "/top/u0" (Black Box Module):
Attribute STATE set to IMPLEMENT.
-------------------------------
```

プロキシ ロジック

パーティションが空になるのを回避するため、NGDBuild によりクロック ポート以外のパーティション ポートにプロキシ ロジック (LUT1) が追加されます。タイミング クロージャのため、最終的にはこのブラック ボックス モジュールを実際のロジックに置換することをお勧めします。

このプロキシ ロジックに、TPSYNC または PPS タイム グループを使用して FROM:TO 制約を適用できます。

デザインによって、これらの制約には次の制約例の一部またはすべてが含まれます。

```
TIMESPEC TS_FFS2PPS = FROM FFS TO PPS 3 ns;
TIMESPEC TS_PPS2FFS = FROM PPS TO FFS 3 ns;
TIMESPEC TS_PPS2PPS = FROM PPS TO PPS 3 ns;
```

次の制約も使用できます。

- ブロック RAM や I/O ロジックなどのその他の同期エンドポイントを使用した TO または FROM 制約
- パーティションごとに PPS タイム グループが作成されたパーティション特定の制約

```
TIMEGRP “PPS_TM1” = PPS(u1/*);
TIMESPEC TS_FFS2PPS_TM1 = FROM FFS TO PPS_TM1 3 ns;
```

このプロキシ ロジックは、次のようになります。

- 実際のロジックのプレースホルダーとして挿入されます。
- 最終デザインには含まれません。

ブラック ボックスがサポートされない状況

専用接続が関係する場合は、ブラック ボックスはサポートされません。

たとえば、パーティション モジュールに、トップ パーティションの専用 I/O サイトに接続されるギガビット トランシーバー (GT) が含まれる場合、このインスタンスをブラック ボックスにすることはできません。

専用接続の両方のエンドポイント (この例の場合は I/O と GT) が同じパーティションに含まれていることが必要です。この例の場合、次のいずれかの方法を使用するとパーティションをブラック ボックスにすることができます。

- GT をトップ パーティションに移動します。
- I/O バッファーおよびパッドを子パーティション内に配置します。

ブラック ボックスを含むデザインの合成とインプリメンテーション

ブラック ボックスを含むデザインを合成およびインプリメントする場合、デザインの一部は存在しません。これにより、予測されない結果が発生することがあります。ブラック ボックスを使用する際は、次の推奨事項に従ってください。

グローバル クロック バッファーを最上位にインスタンシエート

グローバル クロック バッファーを最上位にインスタンシエートします。合成ツールで推論されないようにしてください。グローバル クロック バッファーを最上位にインスタンシエートしない場合、グローバル クロックが使用されないか、クロック ネットに複数のグローバル バッファーが配置される可能性があります。

I/O バッファーを下位パーティションにインスタンシエートしない

I/O バッファーを下位パーティションにインスタンシエートしないでください。I/O バッファーを下位パーティションにインスタンシエートすると、バッファーがトップ パーティションにも推論された場合に、1 つのネットに複数の I/O バッファーが存在することになります。

これらの推奨事項に従うことができない場合は、**`BUFFER_TYPE=NONE`** を使用してバッファー (グローバルまたは I/O) がいつ推論されるかを制御してください。

`BUFFER_TYPE` の詳細は、次を参照してください。

- 『XST ユーザー ガイド (Virtex®-4、Virtex-5、Spartan®-3 および CPLD デバイス用)』(UG627) (付録 A「その他のリソース」にリンクをリスト)
- 『XST ユーザー ガイド (Virtex-6 および Spartan-6 デバイス用)』(UG687) (付録 A「その他のリソース」にリンクをリスト)

パーティションのコンテキスト ルール

「コンテキスト」とは、パーティションの周囲のロジックを表します。

あるコンテキストでインプリメントしたパーティションを別のコンテキストのデザインにインポートすると、エラーが発生する可能性があります。

コンテキストの変更が許容される状況

通常、パーティションのインプリメンテーションとインポートの両方に汎用配線を使用していれば、コンテキストの変更は許容されます。次のような例があるとします。

> インプリメントされたパーティションにレジスタが含まれており、そのレジスタが親に含まれる別のレジスタに接続されています。

インポートすると、パーティション内のレジスタが親パーティションの LUT に接続されます。これらの 2 つのコンテキストに配置または配線の制限はありません。

コンテキストの変更が許容されない状況

次の条件が両方満たされる場合、コンテキストの変更は許容されません。

- パーティション内のロジックがパーティション外のロジックと通信する。
- 汎用配線が使用されていない。

子パーティションの再インプリメント

子パーティションに次のものが含まれていると、コンテキストが変更された場合に子パーティションの再インプリメントが必要になります。

汎用レジスタ

親パーティションの IBUF で駆動されている、**IOB=FORCE** 属性が設定された汎用レジスタ

フリップフロップは IOB にパックされます。インポートすると、このレジスタが親パーティションの別のロジック (LUT など) で駆動され、**IOB=FORCE** が削除されます。

クロッキング モジュール

CLKIN が親パーティションの PLL_ADV で駆動される、DCM_ADV などのクロッキング モジュール

インポートすると、DCM_ADV が PLL 以外のもので駆動されます。これは、親パーティションで CLKIN のドライバーが IBUF (IBUFG、IBUFGDS など) から IBUF 以外のものに変更されるからです。

ISERDES ブロック

親パーティションの IODELAY ブロックで駆動される ISERDES ブロック

インポートすると、ISERDES が IODELAY 以外のもので駆動されます。

専用接続を持つブロック

親パーティションのロジック (IPAD/OPAD) への専用接続 (GT ブロックの RX/TX ピン) を持つブロック

インポートすると、RX/TX ピンが専用 IPAD/OPAD 接続以外のもので駆動されます。ブロックとその専用接続は、1 つのパーティションに含めることをお勧めします。GT ブロックを親パーティションに移動するか、IPAD/OPAD を子パーティションに移動してください。

XILINX®

第 2 章

設計に関する考慮事項

階層デザイン フローを使用するかどうかは、タイミング クロージャでの問題や結果に一貫性がないなどの問題が発生してからではなく、設計の初期段階で決定します。

階層デザインフローの利点を最大限に活用するには、次の事項を考慮してください。

- デザインの論理および物理レイアウト
- HDL コーディングガイドライン
- フロアプラン制約の使用

最適化の制限

パーティションにより境界が作成されるため、最適化に次のような制限が発生します。

- パーティションの境界を越えた最適化は実行されない
- パーティションの入力に供給される定数
- パーティションの未接続の出力は最適化されない
- 1 つのパーティションのロジックを別のパーティションのロジックにパックできない

デザインにパーティションを 1 つ追加しただけで、すべてのインスタンスがパーティションの一部になります。パーティションとして指定されていないインスタンスは、最上位パーティションの一部となります。

パーティションの境界を越えた最適化は実行されない

パーティションの境界を越えた最適化は実行されません。この制限には、次のものの間の最適化が含まれます。

- 親パーティションと子パーティション
- 2 つの子パーティション

フラット デザインと比較した場合、この最適化の制限が次のものに影響します。

- タイミング
- リソース使用率
- 消費電力

次に、最適化の制限の例を示します。

- 1 つのパーティションの組み合わせロジックを別のパーティションの組み合わせロジックと結合して最適化することはできません。
- パーティション間ではリソースの共有は実行されません。

  複数のインスタンスに含まれる共通ロジックを共有して最適化するには、それらのインスタンスを 1 つのパーティションに含める必要があります。

ロジックを特定の組み合わせで使用することにより、ソフトウェアで特定のハードウェア機能を利用したデザインが生成されるようにすることができます。

- ブロック RAM をフリップフロップに直接接続すると、ブロック RAM 専用のレジスタが使用されます。
- DSP をフリップフロップに接続すると、DSP 内にフリップフロップが取り込まれ、より高速のパイプライン DSP が作成されます。

これらのエレメントの間にパーティションの境界があると、これらのパフォーマンス最適化は実行されません。最適化が実行されるようにするには、これらのエレメントを 1 つのパーティションに含める必要があります。

パーティションの入力に供給される定数

最適化で削除されることを目的としてパーティションの入力を定数値に固定している場合、この定数はパーティションの境界を越えてパーティションに挿入できないので、最適化は実行されません。この状態は、コアまたはモジュールの特定の機能をイネーブルまたはディスエーブルにするために定数を使用している場合に発生します。

モジュールのロジックをポートを介して制御することはお勧めしません。次の方法を使用することをお勧めします。

- パラメーターまたは属性を使用する。
- パッケージ ファイルを含める。

パーティションの未接続の出力は最適化されない

パーティションの未接続の出力は最適化されません。パーティションの出力が何も駆動していない場合、ソース ロジックはフラット フローでのようには最適化されません。

1 つのパーティションのロジックを別のパーティションのロジックにパックできない

1 つのパーティションのロジックは、別のパーティションのロジックにはパックできません。この制限は、フリップフロップと LUT の比率が大きく異なる場合にリソース使用率に影響します。パーティション内の組み合わせロジックが最終的にフリップフロップになる出力を駆動している場合、LUT はフリップフロップと共にパックできません。

パーティションを設定するインスタンスの制限

次のようなインスタンスはパーティションに設定できません。

- モジュールまたはエンティティが、独自の HDL ファイルで定義されていない。
- インスタンス名が変更する可能性がある (インスタンス名がパラメーターまたはジェネレート文に基づいている場合)。

アクティブ Low の制御セット

メモ：次の説明ではリセットを例として使用していますが、同じ概念はクロック イネーブルにも適用されます。

制御ピン (リセットまたはクロック イネーブル) にはローカル インバーターはありません。

パーティションを含まないデザイン

パーティションを含まないデザインでアクティブ Low リセットを使用すると、LUT を使用して信号が反転されます。

- アクティブ Low のリセット

  1 つまたは複数の LUT が推論されます。これらを 1 つの LUT に結合し、I/O エレメントに挿入できます。LUT は削除されます。

- アクティブ High のリセットとアクティブ Low のリセット

  LUT インバーターを 1 つの LUT に結合できます。この LUT はデザインに残りますが、リセット ネットの配線およびタイミングにはそれほど影響しません。LUT 出力は、グローバル リソースに配置できます。

パーティションを含むデザイン

パーティション内でアクティブ Low のリセットを使用するデザインでは、インバーターは次のように処理されます。

- パーティション内で推論できます。
- パーティション外に取り出すことはできません。
- 親パーティションのほかのインバーターと結合することはできません。

この場合、次のようになります。

- リセットはグローバル リソースに配置できません。
- デザインが密集している場合、リセットのタイミングおよび配線の問題が発生する可能性があります。

この問題を回避するには、アクティブ Low の制御信号を使用しないでください。ただし、AXI インターフェイスを使用する IP コアの場合など、それが不可能な場合もあります。その場合、次のようにすることをお勧めします。

- アクティブ Low のリセットを最上位信号に割り当てます。
- その信号をデザインのその他の場所で使用します。

  例：`reset_n <= !reset;`

  - すべてに **`reset_n`** を使用します。
  - 上記以外の信号またはポートの割り当てには **`!reset`** は使用しません。

このようにすると、次の利点があります。

- デザイン全体でリセット ネットに LUT のみが推論されます。
- デザイン パフォーマンスへの影響は最小限に抑えられます。

BoundaryOpt 属性を使用した IP コアの最適化

IP コアを最適化するには、パーティションで **BoundaryOpt** 属性を使用します。

BoundaryOpt の詳細は、次を参照してください。

- 第 4 章「コマンド ラインでのパーティション フロー」
- 第 5 章「PlanAhead でのパーティション フロー」

BoundaryOpt の利点

BoundaryOpt 属性には、次のような利点があります。

- パーティションのインターフェイスが緩和されます。
- 次の一部の最適化が可能になります。
  - 入力/出力定数
  - 未接続出力

BoundaryOpt の制限

BoundaryOpt には、次のような制限があります。

RTL アクセスのないモジュール

BoundaryOpt は、IP コアなど、レジスタ トランスファー レベル (RTL) アクセスのないモジュールでのみ使用してください。RTL アクセスのあるモジュールでは、HDL コードでこれらのインターフェイスの問題を解決する必要があります。

インプリメンテーションでのパーティションに対する BoundaryOpt の設定

インプリメンテーションでパーティションに **BoundaryOpt** を設定すると、パーティションのインターフェイスが変更される場合があります。たとえば、パーティション ポートが最適化で削除されることがあります。

これらの最適化されたポートにロジックを接続するために親パーティションも変更された場合、次のようになります。

- パーティション インターフェイスはインプリメンテーション実行からエクスポートされたデータと一致しなくなります。
- エクスポートされたデータは失われます。
- 親パーティションと子パーティションの両方を再インプリメントする必要があります。

これは、次の 2 つの規則にまとめられます。

- 各実行でパーティション インターフェイスのコンテキストを同じにする必要があります。**BoundaryOpt** を使用した場合、ポートが最適化で削除されるなど、この条件が満たされなくなる場合があります。
- **BoundaryOpt** の値は、各実行で同じにする必要があります。

最適化の制限

BoundaryOpt により、最適化できるものとできないものがあります。次の図に、**BoundaryOpt** で最適化される場合を示します。

定数のプッシュ

図 2-1 「定数のプッシュ」では、**BoundaryOpt** により定数がパーティションの境界を 1 つだけ越えてプッシュされ、パーティション インターフェイスからポートが削除されます。ルート スルーネットは削除されません。

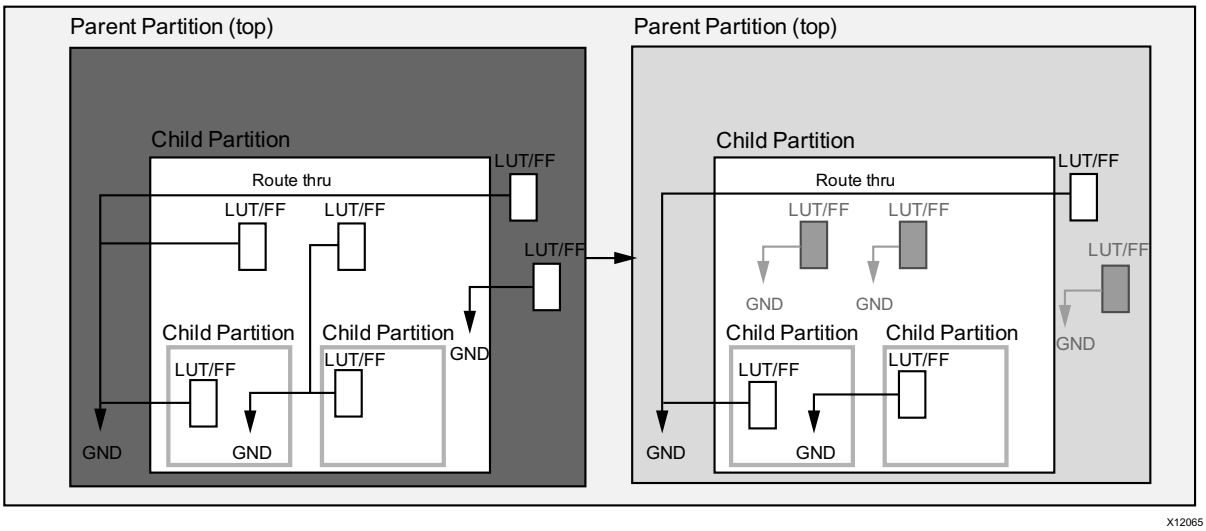

図 2-1 : 定数のプッシュ

未使用の出力

図 2-2「未使用の出力」では、**BoundaryOpt** により未使用のパーティション出力が切断され、パーティション インターフェイスからポートが削除されます。

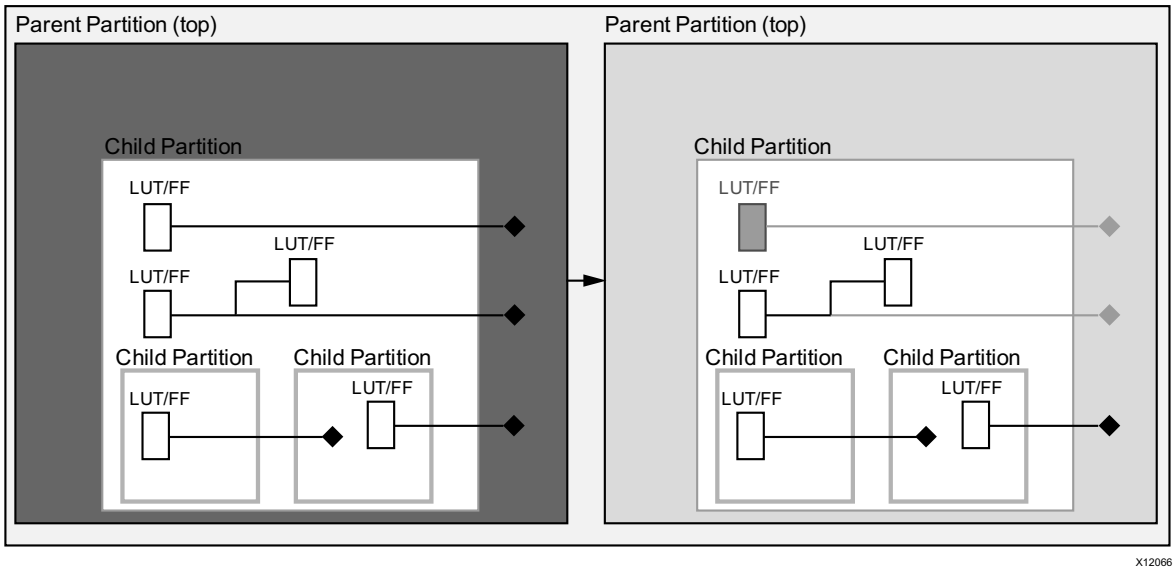

図 2-2 : 未使用の出力

BoundaryOpt の値

BoundaryOpt には、次の値を設定できます。

- **all**
- **none**

パーティションに設定された値は、マップ レポートのパーティション サマリのセクションにリストされます。

デザインの構造

デザインをフラット フローを使用してインプリメントすると、合成ツールおよびインプリメンテーション ツールでデザイン全体をスピードおよびエリアを優先して最適化できます。デザインは階層の境界を越えて最適化されるので、デザインの論理レイアウトはそれほど重要ではありません。

デザインを階層フローでインプリメントする場合、ロジックが分離されるので、パーティションが最適化の壁となり、デザインに悪影響を及ぼすことがあります。次の図に例を示します。

デザイン階層の例

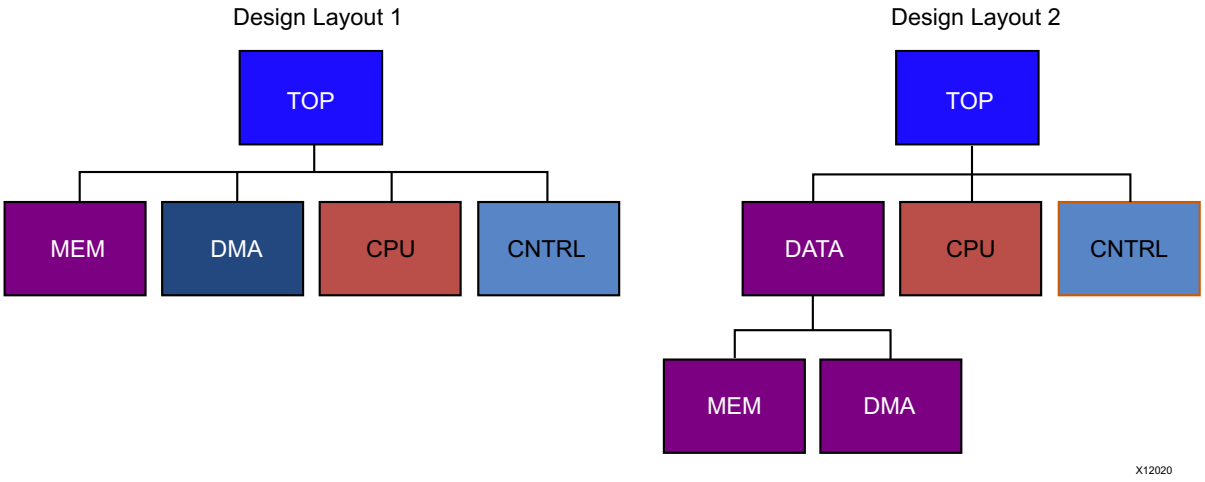

図 2-3 : デザイン階層の例

デザイン レイアウト 1 での最適化

デザイン レイアウト 1 には、同じ階層レベルに **MEM** および **DMA** というモジュールがあります。

TOP の下にあるすべてのモジュールにパーティションを追加すると、**MEM** と **DMA** の間では最適化は実行されません。

これらの2 つのモジュールに関連するロジックが多数ある場合、次のようになります。

- フラットフローでは、モジュールの最適化が実行されます。
- 階層デザインフローでは最適化は実行されず、リソース使用量が増加し、よいタイミング結果が得られない可能性があります。

デザイン レイアウト 2 での最適化

デザイン レイアウト 2 では、共有ロジックを含むモジュールが 1 つのパーティション **DATA** の下にグループ化されており、フラット フローと同様に **MEM** と **DMA** の間で最適化が実行されます。

階層デザイン フローの有効活用

階層デザイン フローを使用するかどうかは、デザインの構築、モジュール インターフェイスの定義、およびモジュール コードの記述の前に決定する必要があります。フラット フローでタイミング問題が発生した後に、タイミング クロージャ手法として階層デザイン フローを使用しないでください。

パーティションを使用するとモジュールが分離され、ソフトウェアで境界を越えた最適化は実行されなくなります。

階層デザイン フローの利点を活かすため、次のようなガイドラインおよび推奨事項があります。

- 入力ポートおよび出力ポートにレジスタを付ける
- パーティションの内部と外部の両方にネットを使用しない
- ファンアウトの大きいネットを管理する
- 定数をネットとして使用しない
- ポートを未接続のままにしない
- 専用接続を 1 つのパーティション内に配置する

入力ポートおよび出力ポートにレジスタを付ける

可能な限り入力および出力にレジスタを付けます。

パーティションの境界を越えた最適化は実行されないので、入力および出力でタイミングの問題が発生しやすくなります。

入力および出力にレジスタを付けると、次のような利点があります。

- パーティション内のパスに焦点を置いた処理が可能となります。
- モジュールのタイミングを確実に保持できます。

パーティションの境界を横切るネットは、ネットに接続されているパーティションすべてをインポートしない限り保持されません。そのため、タイミング クリティカル ネットがこの境界を横切る場合、インポートした場合でもパーティションでタイミング違反が発生する可能性があります。このネットにレジスタを付けると、パーティション内でタイミング クリティカルな変更が発生するのを回避できます。

パーティションの内部と外部の両方にネットを使用しない

パーティションの内部と外部の両方にネットを使用しないようにします。

ネットをパーティション内および出力ポートとして使用する必要がある場合は、次のようにします。

- ネットのソースを複製します。
- 1 つのネットを内部ネットとして使用します。
- もう 1 つのネットを出力ポートとして使用します。

ファンアウトの大きいネットを管理する

パーティション モジュールの出力のファンアウトも考慮する必要があります。パーティションの出力のファンアウトが大きく、デザインの複数のエリアに接続される場合、ドライバーを複製する必要がある場合もあります。

- フラット フローでは、この複製は自動的に実行されます。
- 階層デザインフローでは手動で複製する必要があります。

定数をネットとして使用しない

定数をネットとして使用しないようにします。

IP コアなどのネットリスト モジュールには、ロジックの不要な部分がマップの最適化で削除されることを前提として設計されているものがあります。このような IP では、コアの特定のロジックをイネーブルまたはディスエーブルにするために、入力を定数に接続します。

この方法には、次のような特徴があります。

- IP をネットリストとして提供すると同時に、多少のカスタマイズも可能になります。
- パーティションでは境界を越えた最適化が実行されないので、機能しません。

ポートに定数を接続することにより最適化でロジックが削除されることを前提とした EDIF/NGC コアにパーティションを直接追加すると、ロジックの最適化は実行されず、最適な結果は得られません。

このように動作するコアにパーティションを追加する必要がある場合は、次のようにしてください。

- EDIF または NGC コアを含む HDL ラッパーを追加します。
- このラッパーにパーティションを配置します。

HDL ラッパーのポート リストには、IP コアをデザインの残りの部分に接続するのに必要な I/O のみを含めるようにし、定数の割り当てはラッパー ロジック内に残します。パーティションを IP コアの上のレベルで定義することにより、マップ ツールで定数をコアにトレースすることが可能となり、必要な最適化が実行されます。

ポートを未接続のままにしない

ポートを未接続のままにしないようにします。

未接続のネットでも、同じような最適化の問題が発生します。パーティションの出力を未接続のままにすると、ソースのないこのネットに接続されたドライバーがマップの最適化で削除されません。

このロジックを HDL コードで削除できない場合は、パーティションを含むラッパーを作成すると次のようになります。

- パーティションの境界が移動します。
- マップ ツールでロジックを最適化できるようになります。

パーティションの出力が次のすべての条件に一致する場合、エラーが発生することがあります。

- ロジックで駆動されていない。
- パーティションの外部のロジックに接続されいてる。

インプリメンテーション ツールでパーティションの出力が駆動されていないことを検出できず、部分的なネットが配線されます。これは無効であり、BitGen でデザイン ルール チェック (DRC) エラーが発生します。不要なパーティション ポートを削除し、パーティションを再インプリメントしてください。

専用接続を 1 つのパーティション内に配置する

専用接続を 1 つのパーティション内に配置します。

FPGA デバイスのエレメントには、特定の機能や高速の低スキュー配線用の専用接続を提供するため、特定の方法で連動するものがあります。

これらのエレメントが別々のパーティションに含まれていると、正しくコンフィギュレーションされません。これらのインスタンスは 1 つのパーティション内に配置します。

次に、これらのエレメントをリストします。

- OSERDES、IODELAY、および OBUFTDS
- OSERDES、ODDR、および OBUFTDS
- IDELAY および IDELAYCNTRL
- ISERDES、IDDR、および IBUFDS

配線情報の保持に関する制限

すべての配線情報を保持できないことがあります。

次のような場合、パーティションの配線が再配線される可能性があります。

- インポートされたパーティションの保持レベルが、配線ではなく配置または合成に設定されている。
- 下位パーティションの I/O バッファーがインポートされ、接続されているパッドがインプリメントされる親パーティションに含まれている。

  これは専用配線であるため、厳密にいえば保持されませんが、変更されません。
- 下位パーティションのフリップフロップがインポートされ、インプリメントされる親パーティションに含まれている I/O バッファーに接続されている。

  これは専用配線であり、フリップフロップが I/O ロジックにパックされれば変更されません。フリップフロップをパーティションの境界を越えて取り込み、I/O ロジックにパックするため、**IOB=FORCE** または **IOB=TRUE** 制約を設定する必要があります。フリップフロップがスライスにパックされている場合、配線は保持されず、タイミングが満たされるとは限りません。

  フリップフロップが I/O ロジックにパックされない場合、次のようになります。

  - **IOB=FORCE** を使用した場合はエラーが表示されます。
  - **IOB=TRUE** を使用した場合は警告が表示されます。
- 下位パーティションの LUT がインポートされ、インプリメントされる親パーティションに含まれている I/O バッファーに接続されている。このインスタンスは、次のようになります。
  - LUT はスライス ロジックにパックされる必要があります。
  - 専用配線ではありません。
  - タイミングが保持されるとは限りません。
- デザインに PWR または GND ネットが含まれ、トップ パーティションがインプリメントされる。

  PWR および GND ネットは常にインプリメントされ、子パーティションに含まれていても、最上位パーティションと共にインポートされます。

パーティションのフロアプラン

メモ：このセクションでは、AREA_GROUP 制約を使用する方法を説明します。

フロアプランとは、デザインの配置を制約を使用して制御することです。

CLB 境界にスライス範囲を作成する

パーティション デザインで AREA_GROUP 制約を使用するのに制限はありませんが、CLB 境界にスライス範囲を作成することをお勧めします。これにより、配置および配線のリソースを最大限に利用できます。

スライス範囲が CLB 境界上にあるかどうかを検証するには、制約の XY 座標を確認します。

XY 座標が次のようになっている場合、アドレスは CLB 境界にあります。

- 偶数で開始している。
- 奇数で終了している。

  例 : **`X0Y0 ～ X3Y9`**

PlanAhead™ ソフトウェアを使用して、次を実行できます。

- AREA_GROUP 制約を作成します。
- 制約を CLB 境界上に自動的に設定します。

CLB およびその他のブロックの詳細は、付録 A「その他のリソース」にリストされているデバイスのデータシートを参照してください。

ロジックを 1 つのエリアに制限する

パーティションでフロアプランを使用すると、パーティションに関連するすべてのロジックをデバイスの 1 つのエリアに制限できます。

ロジックをデバイスの 1 つのエリアに制限すると、次の利点があります。

- 各パーティションを配置配線する領域を作成します。
- インポートの際に配線の競合が発生する可能性を最小限に抑えます。
- 後で追加するロジック用にデバイスのその他の部分を予約できます。

コマンド ライン ツールの使用

パーティションは PlanAhead ツールでサポートされていますが、コマンド ライン ツールを使用してパーティション デザインを実行することも可能です。

この場合、PlanAhead ソフトウェアを使用して次を実行できます。

- パーティションの設定
- デザインのフロアプラン
- PXML ファイルの作成

詳細は、第 4 章「コマンド ラインでのパーティション フロー」を参照してください。

デザイン保持

デザイン保持パーティションでは AREA_GROUP 制約は必要ありませんが、デザインによっては次のような利点があります。

- ランタイムとタイミング結果が向上します。
- インポートの際に配置または配線の競合が発生する可能性を低減します。

チーム デザイン フローでの要件

- 各チーム メンバー パーティションには、そのパーティションに必要なロジックを含む AREA_GROUP RANGE 制約を設定する必要があります。

  チーム メンバー パーティションを AREA_GROUP RANGE 外に配置する必要がある場合は、そのロジックをトップ パーティションに移動してアセンブリ中の配置の競合を回避してください。

- あるチーム メンバー パーティションの AREA_GROUP RANGE をほかのチーム メンバー パーティションの AREA_GROUP RANGE とオーバーラップさせることはできません。

  これは、トップ パーティションの AREA_GROUP RANGE 制約でも同じです。

- 各チーム メンバー パーティションは、次のものに含める必要があります。
  - 1 つの AREA_GROUP
  - その AREA_GROUP の子
- 1 つのチーム メンバーのロジックとほかのチーム メンバーのロジックを 1 つの AREA_GROUP 制約に含めないでください。2 つのチーム メンバー パーティションの間のインターフェイス タイミングにクリティカルであるロジックは、トップ パーティションに移動します。このようにすると、ロジックを両方のチーム メンバー パーティションにまたがって配置できます。

プロセッサ システムのパーティション

EDK プロセッサ システムにパーティションを設定する際は、最初の実行およびその後の実行で、NGDBuild に BMM ファイルを入力する必要があります。

第 3 章

合成パーティションフロー

階層デザイン フローでは、各パーティションが個別に合成されます。

各パーティションを個別に合成することにより、1 つのエリアが変更されたときに別のエリアの合成結果が変更されるのを回避できます。

次の合成パーティション フローを使用できます。

- インクリメンタル合成パーティション フロー
- ボトムアップ合成パーティション フロー

インクリメンタル合成パーティション フロー

ほとんどのサードパーティ合成ツールで、インクリメンタル合成がサポートされています。

インクリメンタル合成では、次のように処理されます。

- レジスタ トランスファー レベル (RTL) モジュールをパーティションとして設定できます。
- 各パーティションは個別に合成され、階層の境界が作成されます。
- 1 つのモジュールで HDL が変更されても、ほかのモジュールには影響しません。
- 再合成するモジュールまたはインスタンスは、HDL または制約の変更に基づいてツールにより判断されます。

サポートされるインクリメンタル合成パーティション フロー

サポートされているインクリメンタル合成フローは、次のとおりです。

- Synplify Pro および Synplify Premier

  コンパイル ポイントを使用します。
- Precision

  HDL の属性を使用してパーティションを指定します。

インクリメンタル合成パーティション フローの利点

インクリメンタル合成パーティション フローには、次のような利点があります。

- 各パーティションに対して個別の合成プロジェクト ファイルを作成する必要はありません。
- フラット合成フローからの移行が簡単です。
- 再合成が必要なモジュールは、HDL コードおよびタイミング制約の変更に基づいて合成ツールにより判断されます。

ボトムアップ合成パーティション フロー

ボトムアップ合成パーティション フローには、次のような特徴があります。

- 各パーティションに個別の合成プロジェクトおよびネットリストがあります。
- 再合成する必要のある合成プロジェクトは、HDL コードおよび合成制約の変更に基づいてユーザーが判断します。
- 下位モジュールをブラック ボックスとして最上位を合成します。
- 下位モジュールは、I/O またはクロック バッファーを推論せずに合成します。

サポートされるボトムアップ合成パーティション フロー

ボトムアップ合成パーティション フローは、次のツールでサポートされます。

- サードパーティ合成ツール
- XST (Xilinx Synthesis Technology)

  XST では、HDL の属性を使用してパーティションを指定します。

ボトムアップ合成パーティション フローの利点

ボトムアップ合成パーティション フローには、次のような利点があります。

- 各チーム メンバーが独自の合成プロジェクトを作成します。

  合成中、複数のチーム メンバーが同じデザインを作業できます。
- 再合成するインスタンスをチーム メンバーが制御します。
  - チーム メンバーが再合成するプロジェクトを決定するので、どのインスタンスが再合成されたかを容易に判断できます。
  - 各ネットリストに個別のタイム スタンプがあります。

合成ツール

次の合成ツールについて説明します。

- Synplify Pro および Synplify Premier
- Precision
- XST (Xilinx Synthesis Technology)

Synplify Pro および Synplify Premier

Synplify Pro およびSynplify Premier でパーティションを使用するには、次の 2 つの方法があります。

- Synplify Pro および Symplify Premier のボトムアップ合成パーティション フロー
- Synplify Pro および Symplify Premier のインクリメンタル合成パーティション フロー

Synplify Pro および Symplify Premier のボトムアップ合成パーティション フロー

Synplify Pro および Symplify Premier のボトムアップ フローでは、各インスタンスに Synplify プロジェクト ファイルがあります。下位プロジェクト ファイルに IOB コンポーネントまたはグローバル クロック バッファーを推論しないようにしてください。

- プロジェクト ファイルで I/O 挿入をオフにするには、次の構文を使用します。

  ```
  set_option disable_io_insertion 1
  ```

- クロック バッファーの使用をオフにするには、**syn_noclockbuf** 属性を使用します。
  - 次の構文を Synplify デザイン制約 (SDC) ファイルに追加します。

    ```
    define_attribute { clock_port } syn_noclockbuf 0
    define_global_attribute  syn_noclockbuf 0
    ```

    または

  - **syn_noclockbuf** を HDL コードに直接挿入します。

詳細は、Symplify のマニュアルを参照してください。

Synplify Pro および Symplify Premier のインクリメンタル合成パーティション フロー

Synplify Pro および Symplify Premier では、インクリメンタル合成パーティション フローがサポートされています。インクリメンタル合成パーティション フローでは、コンパイル ポイントを使用してデザインを小型の合成ユニットに分割します。

locked モードを使用すると、コンパイル ポイントを越えたロジックの移動は実行されません。soft モードおよび hard モードでは境界を越えた最適化が許容されますが、サポートされていません。

パーティションのネットリストが変更された場合、パーティションの再インプリメントが必要になります。インポートされたパーティションがネットリストと一致しない場合、NGDBuild ツールでエラーが発生します。

次に、SDC ファイルのコンパイルポイントの例を示します。

```
define_compile_point {v:controller} -type {locked} -cpfile {}
```

Synplify 合成レポート

SDC ファイルでコンパイル ポイントを作成および変更するには、Synplify を使用してください。合成レポート ファイルに、各コンパイル ポイントのステータスが示されます。

```
Summary of Compile Points
Name Status Reason
-------------------------------------------------
controller Unchanged -
elevator_car Remapped Design changed
express_car Remapped Design changed
top Remapped Design changed
=================================================
```

ザイリンクス インプリメンテーション ツールの実行 (Synplify)

合成が終了したら、次のいずれかの方法でザイリンクス インプリメンテーション ツールを実行します。

- ISE Design Suite コマンド ライン フロー (Synplify)
- PlanAhead ソフトウェア フロー (Synplify)
- Synplify コックピット (Synplify)

ISE Design Suite コマンド ライン フロー (Synplify)

Synplify で Tcl コマンドが実行され、ザイリンクス インプリメンテーション ツールで使用する PXML ファイルが作成されます。

詳細は、第 4 章「コマンド ラインでのパーティション フロー」を参照してください。

PlanAhead ソフトウェア フロー (Synplify)

PlanAhead™ ソフトウェア フローでは、次のものをインポートします。

- 合成ネットリスト
- Synplify で生成された PXML ファイル (オプション)

PlanAhead ソフトウェア内でパーティションを手動で再定義することも可能です。

詳細は、第 5 章「PlanAhead でのパーティション フロー」を参照してください。

Synplify コックピット (Synplify)

Synplify コックピットからザイリンクス インプリメンテーション ツールを実行します。

詳細は、http://www.synopsys.co.jp/ から Synplify Pro および Synplify Premier の資料を参照してください。

Precision

Mentor Graphics 社の Precision 合成ツールでも、階層デザイン フローがサポートされます。最もよく使用されるフローは、パーティション ベースのインクリメンタル フローです。このフローでは、パーティションは属性を使用してモジュールまたはインスタンスに設定します。

Precision 合成レポート

```
Verilog Module:
module my_block( input clk; ...)/* synthesis incr_partition */;
Verilog Instance:
my_block my_block_inst( .clk(clk), ....data_out(data_out) ); // synthesis attribute
my_block_inst
incr_partition true
VHDL Module:
entity my_block is port( clk: in std_logic; ...); attribute incr_partition : boolean;
attribute incr_partition
of my_block : entity is true; end entity my_block;
VHDL Instance:
component my_block is port( ... end component; ... attribute incr_partition : boolean;
attribute
incr_partition of my_block_inst : label is true; ... my_block_inst
```

合成レポートには、パーティションのステートに基づいてパーティションが最適化されたかどうかが示されます。

```
[16027]:Incremental:Skipping Optimize for <...>.fifo_16_64.rtl_unfold_0
[16027]:Incremental:Skipping Optimize for <...>.fir_filter.rtl_unfold_0
[15002]:Optimizing design <...>.fsm.rtl_unfold_0
[16027]:Incremental:Skipping Optimize for <...>.fir_top.rtl
```

ザイリンクス インプリメンテーション ツールの実行 (Precision)

合成が終了したら、次のいずれかの方法でザイリンクス インプリメンテーション ツールを実行します。

- ISE Design Suite コマンド ライン フロー (Synplify)
- PlanAhead ソフトウェア フロー (Precision)
- Precision (Precision)

ISE Design Suite コマンド ライン フロー (Synplify)

[Place & Route] をクリックし、ザイリンクス インプリメンテーション ツールで使用する PXML ファイルを作成します。

詳細は、第 4 章「コマンド ラインでのパーティション フロー」を参照してください。

PlanAhead ソフトウェア フロー (Precision)

PlanAhead ソフトウェアでは、合成済みネットリストをインポートします。パーティションは、PlanAhead ソフトウェアで定義します。

詳細は、第 5 章「PlanAhead でのパーティション フロー」を参照してください。

Precision (Precision)

[Place & Route] をクリックして ISE® Design Suite のインプリメンテーション ツールを実行します。

このフローの詳細は、Mentor Graphics 社のSupportNet サイト http://supportnet.mentor.com/japan/index.cfm からアプリケーション ノートを参照してください。

XST (Xilinx Synthesis Technology)

XST (Xilinx Synthesis Technology) では、次のフローがサポートされています。

- XST ボトムアップ合成パーティション フロー
- XST トップダウン インクリメンタル フロー

XST ボトムアップ合成パーティション フロー

XST ボトムアップ合成パーティション フローでは、パーティションとして使用される各インスタンス (トップ パーティションを含む) に合成プロジェクト ファイルがあります。

> **注意**：最上位パーティションに IOB コンポーネントまたはグローバル クロック バッファーが含まれている場合は、下位パーティションでは推論しないでください。

I/O コンポーネントが推論されないようにするには、XST ファイルで次のオプションを指定します。

```
-iobuf NO
```

下位パーティションに IOB コンポーネントを使用することは可能ですが、その場合は最上位パーティションに余分な IOB が推論されないようにする必要があります。IOB の挿入をオフにするには、次のように設定します。

- 最上位全体に対しては、**-iobuf** オプションを設定します。
- 個々の信号に対しては、信号名に **BUFFER_TYPE=NONE** 属性を設定します。

BUFFER_TYPE=NONE 属性は、下位パーティションにグローバル バッファー (BUFG) がインスタンシエートされている場合に、最上位パーティションで BUFG が推論されないようにするためにも使用できます。

BUFFER_TYPE 属性の設定方法、XST をコマンド ライン モードで実行する方法の詳細は、次を参照してください。

- 『XST ユーザー ガイド (Virtex®-4、Virtex-5、Spartan®-3 および CPLD デバイス用)』(UG627) (付録 A「その他のリソース」にリンクをリスト)
- 『XST ユーザー ガイド (Virtex-6 および Spartan-6 デバイス用)』(UG687) (付録 A「その他のリソース」にリンクをリスト)

XST トップダウン インクリメンタル フロー

XST トップダウン インクリメンタル フローには、次の特徴があります。

- ISE Design Suite 13.1 リリースの新機能です。
- Virtex-6、Spartan-6、および 7 シリーズ FPGA デバイスのみをサポートします。
- Virtex-4、Virtex-5、および Spartan-3 デバイスはサポートされません。
- ボトムアップ フローよりもセットアップが簡単です。
- コマンド ラインおよび PlanAhead ソフトウェアの両方がサポートされています。
- 階層インスタンスにパーティションを定義できます。各パーティションに個別の NGC ファイルが作成されます。

PXML ファイル

パーティションは、`xpartition.pxml` という PXML ファイルで定義および制御します。

- コマンド ラインでは、インプリメンテーションに使用される PXML ファイルと同じまたは異なる PXML ファイルを使用できます。
- PlanAhead ソフトウェアで PXML ファイルを管理できます。

デザイン保持やチーム デザインなどの特定の手法でこの機能を使用する方法については、このガイドのほかの章を参照してください。

XST では、PXML ファイルは次のように読み込まれます。

- PXML ファイルが現在の作業ディレクトリに存在する場合は、自動的に読み込まれます。
- PXML が異なるディレクトリ (インプリメンテーション ディレクトリなど) に存在する場合は、**`-partition_file`** オプションを使用してファイルを指定します。

  このオプションでは、PXML ファイルを検索する複数のリモート ロケーションを指定できます。

変更されていないパーティションのインポート

XST インクリメンタル フローでは変更されていないパーティションをインポートできるので、一部のパーティションが **`Import`** に設定されており、**`ImportLocation`** 属性が定義されている場合があります。**`ImportLocation`** が相対パスで定義されている場合、そのパスは現在の作業ディレクトリに対するパスではなく、PXML ファイルに対するパスである必要があります。

合成が **`import`** に設定されているパーティションで、エクスポートされたインプリメンテーション データが次の場合、**`implement`** に設定する必要がある場合があります。

- 最新でない
- 存在しない

この場合、合成とインプリメンテーションに別々の PXML ファイルを使用した方が簡単である可能性があります。

前回の合成結果をインポートするには、結果を別の場所にエクスポートする必要があります。これにより、次の合成実行で結果が上書きされるのを防ぐことができます。エクスポートされた合成結果には、パーティションの NGC ファイルと HDX ファイルが含まれている必要があります。

XILINX®

第 4 章

コマンド ラインでのパーティション フロー

ISE® Design Suite コマンド ライン パーティション フローは、次のように実行します。

- 合成後のデザインにパーティションを使用します。
- コマンド ラインで実行します。
- ザイリンクス インプリメンテーション ツールを使用します。

ザイリンクス インプリメンテーション ツール

ISE Design Suite コマンド ライン パーティション フローは、次のザイリンクス インプリメンテーション ツールを使用します。

- NGDBuild
- MAP
- PAR

ザイリンクス インプリメンテーション ツールは、現在の作業ディレクトリで PXML ファイルを検索します。現在の作業ディレクトリは、インプリメンテーション ツールを実行したディレクトリです。

ザイリンクス インプリメンテーション ツールは、PXML ファイルから次の情報を取得します。

- デザイン内で定義されているパーティション
- パーティションのステート

PXML ファイルの作成

PXML ファイルとは、次のようなファイルです。

- パーティションの定義が含まれます。
- 大文字と小文字が区別されます。
- 必ず `xpartition.pxml` という名前を付ける必要があります。
- 次のすべてのインプリメンテーション ツールに対して存在している必要があります。
  - NGDBuild
  - MAP
  - PAR

PXML ファイルは、次の方法で作成できます。

- テキスト エディターで手動で作成

  手動で PXML ファイルを作成する際の参考に、PXML ファイルのテンプレートが提供されています。テンプレートファイルは、次のディレクトリにあります。

  `<Xilinx_13_directory>/PlanAhead/testcases/templates/xpartition.pxml`

- PlanAhead™ ソフトウェアなどのソフトウェア ツールで作成

  PlanAhead ソフトウェアで PXML ファイルを作成する方法は、第 5 章「PlanAhead でのパーティション フロー」の「PXML ファイルのエクスポート」を参照してください。

PXML ファイルを手動で作成した場合やサードパーティの合成ツールで生成した場合は、「インプリメンテーションの実行」を参照してください。

下位パーティションと共に、デザインの最上位モジュールをパーティションとして定義する必要があります。子パーティションまたはネストされたパーティションもサポートされています。

PXML ファイルが現在の作業ディレクトリに含まれていれば、インプリメンテーション ツールで認識されます。

PXML ファイルの例

```
<?xml version="1.0" encoding="UTF-8" ?>

<Project FileVersion="1.2" Name="Example" ProjectVersion="2.0">
  <Partition Name="/top" State="implement" ImportLocation="NONE">
    <Partition Name="/top/module_A" State="import" ImportLocation="/
home/user/Example/export" Preserve="routing">
    </Partition>
    <Partition Name="/top/module_B" State="import" ImportLocation="../
export" Preserve="routing">
    </Partition>
    <Partition Name="/top/module_C" State="implement"
ImportLocation="../export" Preserve="placement">
    </Partition>
  </Partition>
</Project>
```

PXML の属性とその値

次の表に、上記の PXML ファイルの例で使用されている属性とその値を説明します。

表 4-1 : Project を定義するための PXML 属性

属性名	値	説明
FileVersion	1.2	ツールで使用されます。この値は変更しないでください。
Name	プロジェクト名	プロジェクト名を指定します。
ProjectVersion	2.0	ツールで使用されます。この値は変更しないでください。

表 4-2 : Partition を定義するための PXML 属性

属性名	値	説明
Name	パーティション名	パーティションを適用するモジュールの階層インスタンス名を指定します。
State	**implement**	パーティションは再インプリメントされます。
	import	パーティションはインポートされ、**Preserve** で設定されたレベルでインプリメンテーションが保持されます。
	auto	論理的に変更されていないパーティションをインポートし、論理的に変更されているパーティションをインプリメントします。**ImportLocation** 属性を設定する必要があります。コマンド ラインでのみサポートされます。
ImportLocation	パス	インポート元のディレクトリを指定します。**State** が **import** に設定されていない場合は無視されます。**State** が **import** に設定されている場合、パスを相対パスまたは絶対パスで指定できますが、指定した場所に export ディレクトリが含まれている必要があります。この属性が無視される場合、ディレクトリを設定したりこの属性を削除する代わりに、**NONE** キーワードを設定できます。
ImportTag	パーティション名	パーティションをインプリメントされた階層レベルとは別の階層レベルにインポートできるようにします。値をインプリメントされたパーティションの階層インスタンスに設定します。
Preserve	**routing**	コンポーネントの配置配線が 100% 保持されます。トップ パーティションのデフォルト設定です。
	placement	配置が保持されます。配線は変更される場合があります。
	synthesis	配置および配線が変更される場合があります。
	inherit	親パーティションの設定値が使用されます。トップ パーティション以外のすべてのパーティションのデフォルト設定です。
BoundaryOpt	**all**	定数に接続されているパーティション ポートおよび未使用のパーティション ポートの最適化をイネーブルにします。
	none	通常のパーティション最適化規則が適用されます。パーティションの境界内での最適化のみを有効にします。これがデフォルト値です。

インプリメンテーションの実行

PXML ファイルを作成したら、ザイリンクス インプリメンテーション ツールをコマンド ラインまたはバッチ モードで実行できます。

基本的なスクリプトは、次のようになります。

```
ngdbuild -uc design.ucf design.edn design.ngd
map -w design.ngd -o design_map.ncd design.pcf
par -w design_map.ncd design.ncd design.pcf
```

NGDBuild コマンドで指定するネットリストには、フラット合成の出力は指定できません。合成プロジェクトを設定および実行する際に、パーティションを考慮する必要があります。詳細は、第 3 章「合成パーティションフロー」を参照してください。

インプリメンテーション ツールのレポート

インプリメンテーション ツールのレポートで、次を確認します。

- ツールで PXML ファイルが認識されている。
- パーティションのステートが正しく設定されている。

NGDBuild レポートの例

NGDBuild レポート (BLD) には、次に示すようなパーティション インプリメンテーション ステータスを示すセクションがあります。MAP レポートおよび PAR レポートにも、同様のパーティションセクションがあります。

```
-------------------------------
Partition Implementation Status
-------------------------------
Preserved Partitions:

Implemented Partitions:

   Partition "/top":
   Attribute STATE set to IMPLEMENT.

   Partition "/top/Control_Module":
   Attribute STATE set to IMPLEMENT.

   Partition "/top/Express_Car":
   Attribute STATE set to IMPLEMENT.

   Partition "/top/Main_Car":
   Attribute STATE set to IMPLEMENT.

   Partition "/top/Tracking_Module":
```

パーティションでサポートされていないインプリメンテーション オプション

パーティションでは、次のインプリメンテーション オプションはサポートされていません。

- **-glob_opt** (グローバル最適化)
- **-smartguide** (SmartGuide™ テクノロジ)
- **-power** (消費電力の最適化)
- **-mt** (マルチスレッド)

ザイリンクス環境変数

インプリメンテーション ツールの動作に影響を与えるザイリンクス環境変数 (XIL_*) も多数あり、ISE Design Suite のリリースによってその影響は異なります。

ザイリンクスの階層デザイン フローは環境変数に対してテストされていないので、ツールを実行する前にザイリンクス環境変数をすべて削除することをお勧めします。

パーティションのエクスポート

満足するインプリメンテーション結果が得られたら、パーティションをエクスポートします。パーティションをエクスポートするには、インプリメンテーション ファイルをエクスポート ディレクトリにコピーします。

インプリメンテーション ディレクトリの完全なコピーのサイズが大きすぎる場合は、スクリプトを使用して、次にのファイルのみをコピーします。

- 必須ファイル
- オプションのファイル

必須ファイル

必須ファイルは、次のとおりです。

- PREV ファイル
- PXML ファイル

PREV ファイル

PREV ファイル (***prev*.***) には、パーティションをインポートするのに必要なインプリメンテーション データが含まれています。パーティションを含むデザインをインプリメントすると作成されます。

下のファイル名には、最上位デザインの名前 **top** が含まれています。

PlanAhead ソフトウェアでは、次の必須 PREV ファイルがプロモートされます。

- top_prev_built.ngd
- top_prev_mapped.ncd
- top_prev_mapped.ngm
- top_routed_prev_routed.ncd

PXML ファイル

パーティションをインポートするのが適切であるかどうかを確認するために必要です。たとえば、PXML ファイルにインポートするパーティションが含まれていない場合、エラーが生成されます。

オプションのファイル

オプションのファイルは、次のとおりです。

- レポート ファイル
- ユーザー制約ファイル (UCF)

レポート ファイル

レポート ファイルの保存はオプションですが、保存しておくと後で参照できるので便利です。

PlanAhead ソフトウェアでは、次のレポート ファイルがプロモートされます。

- NGDBuild (`.bld`)
- Map (`.mrp` および `.map`)
- PAR (`.par`)
- TRACE (`.twr` および `.twx`)

ユーザー制約ファイル (UCF)

ユーザー制約ファイル (UCF) はオプションです。

オプションですが、重要なファイルです。これらのファイルには、デザインをインプリメントしたときに使用されたオプションが記録されています。

> **注意** : エクスポート ディレクトリのファイルは変更しないでください。これらのファイルは、これらのファイルは、ソース ファイルのようなものと考えることができます。

デザインに複数のパーティションがある場合、すべてがエクスポートされます。特定のパーティションがエクスポートされるというよりは、デザイン全体がエクスポートされると考えるとわかりやすいです。

パーティションのステートを import に変更

パーティションをエクスポートしたら、PXML ファイルでパーティションのステートをアップデートする必要があります。これには、テキスト エディターを使用します。

エクスポートしたパーティションをインポートするには、次の手順に従います。

1. パーティションのステートを **`import`** に変更します。
2. **`ImportLocation`** 属性をエクスポートされたパーティションの場所に設定します。

   この属性は、次のいずれかで指定します。

   - 次のような相対パス

     `../export`

   - 次のような絶対パス

     `/home/user/Example/export`

デザインでの反復作業

必要に応じてデザインを変更し、再合成します。合成でエクスポートされたパーティションが変更された場合は、PXML ファイルでそのパーティションのステートを import から implement に手動で変更する必要があります。

インポート ディレクトリにあるパーティションに完全に一致しないパーティションをインポートしようとすると、NGDBuild で次のようなエラー メッセージが表示されます。

```
ERROR:NgdBuild:1037 - The logic for imported partition '/top/Main_Car'
using  previous implementation './export/top_prev_built.ngd' has been
modified.
```

このエラー メッセージは、パーティション ソースが変更されたパーティションを再インプリメントおよびエクスポートしていない場合に表示されます。パーティションを再インプリメントし、エクスポートしてから、デザインにインポートする必要があります。

HDL にコメントを追加した場合にも、ネットリストで多少のロジックの変更や名前の変更が発生する可能性があり、パーティションを再インプリメントする必要があることに注意してください。

次の 2 つの条件が満たされる場合、実行でその結果が使用されるようパーティションをエクスポートする必要があります。

- 保持レベルを routing 以外に設定してパーティションをインポートした。
- PAR ツールにより配置および配線が変更された。

パーティションを含むデザインでの SmartXplorer の使用

パーティションを含むデザインのタイミング要件が厳しい場合、タイミングを満たすために SmartXplorer を使用できます。

1. デザインの完了したパーティション部分に対して SmartXplorer を実行します。
2. 変更される可能性のあるデザインのほかの部分のインプリメンテーションには、通常のフローを使用します。
3. タイミング クリティカルなモジュールの SmartXplorer での結果をインポートします。

SmartXplorer の詳細は、付録 A「その他のリソース」にリストされている『コマンド ライン ツール ユーザー ガイド』(UG688) を参照してください。

ネットリストまたは NGD ファイルを指定して SmartXplorer を実行

SmartXplorer を次のいずれかのファイルを指定して実行できます。

- ネットリスト ファイル (EDIF または NGC)
- NGD ファイル

ネットリスト ファイルまたは NGDファイルと同じディレクトリに PXML ファイルを配置しておく必要があります。

PXML ファイルは通常 NGDファイルと同じディレクトリに配置されているので、次の手順で実行することをお勧めします。

1. NGDBuild を実行します。
2. SmartXplorer を NGD ファイルを使用して実行します。

ネットリスト ファイルを使用する場合は、PXML ファイルをネットリスト ディレクトリにコピーする必要があります。

SmartXplorer のディレクトリ パス

SmartXplorer の実行結果からパーティションをインポートする場合は、PXML ファイルの ImportLocation として SmartXplorer で作成されたディレクトリを指定する必要があります。

ディレクトリ パスは、次のように指定できます。

- 次のような相対パス

  `../run2`

- 次のような絶対パス

  `/home/user/Example/run2`

SmartXplorer の入力ファイル

SmartXplorer には、最上位ネットリストまたは NGD ファイルを入力します。

NGD ファイルを指定する場合は、次の点に注意する必要があります。

- NGD ファイルは、MAP ツールおよび PAR ツールで使用されるのと同じ PXML ファイルを使用して作成する必要があります。

  NGDBuild を実行したときに PXML ファイルが存在しない場合は、MAP および PAR で PXML ファイルが無視されます。

- SmartXplorer で MAP ツールおよび PAR ツールが実行され、NGD ファイルは 1 レベル上に作成されます。

  SmartXplorer の結果をインポートするには、1 レベル上にある `*prev_built.ngd` ファイルを SmartXplorer で作成された適切な `run` ディレクトリにコピーする必要があります。

パーティションの削除と復元

パーティションを削除したり復元したりできます。

個々のパーティションの削除

個々のパーティションを削除するには、次の手順に従います。

1. PXML ファイルからそのパーティションの参照を削除します。
2. 親パーティションのステートを implement に設定します。

すべてのパーティションの削除

すべてのパーティションを削除してフラット フローを実行するには、implementation ディレクトリの PXML ファイルを削除するか、名前を変更します。

パーティションの復元

パーティションを復元するには、次のいずれかを実行します。

- すべてのパーティションが削除されている場合は、PXML ファイルを implementation ディレクトリに戻します。
- 個々のパーティションが削除されている場合は、PXML ファイルにパーティションを定義する行を戻します。

一度削除して復元したパーティションを、インポートできる場合があります。

次の場合にインポートできます。

- 入力ネットリストが変更されていない。
- エクスポート ディレクトリが存在している。

第 5 章

PlanAhead でのパーティション フロー

PlanAhead™ ソフトウェアでは、レジスタ トランスファー レベル (RTL) プロジェクトとネットリスト プロジェクトでパーティション フローがサポートされます。

PlanAhead でのパーティション フローについて

このセクションでは、PlanAhead のパーティション フローについての概要を示します。

パーティションの追加と削除

パーティションを追加または削除するには、次の表示を使用します。

表 5-1 : パーティションの追加と削除を実行する表示

プロジェクト	表示
RTL	RTL Design
ネットリスト	Netlist Design

RTL プロジェクト

RTL プロジェクトには、次のような特徴があります。

- Virtex®-6、Spartan®-6、および 7 シリーズ FPGA デバイスのみをサポートします。
- XST (Xilinx Synthesis Technology) のインクリメンタル機能を使用して、次を実行します。
  - デザインを合成します。
  - 各パーティションに個別の NGC ファイルを生成します。

別の合成ツールまたは XST のボトムアップ合成を使用する場合は、次のようにする必要があります。

- PlanAhead ソフトウェア外で合成を実行します。
- PlanAhead でネットリスト プロジェクトを作成してインプリメンテーションを実行します。

ネットリスト プロジェクトでは、独自のネットリストを持つ階層にのみパーティションを設定します。

フラット ネットリスト プロジェクト

PlanAhead ソフトウェアでフラット ネットリスト プロジェクトを作成し、プロジェクトの階層にパーティションを追加できます。

合成で生成されるネットリストは 1 つだけなので、デザインの 1 つの部分を変更するとほかの部分にも影響し、プロモートした結果をインポートできません。

フラットなグローバル最適化合成フローはサポートされていません。

パーティション用に合成プロジェクトを設定する方法の詳細は、第 3 章「合成パーティション フロー」を参照してください。

新規プロジェクトの作成

PlanAhead ソフトウェア で新規プロジェクトを作成するには、次の手順に従います。

1. Getting Started ページの [Create a New Project] リンクをクリックします。
   開いた New Project ウィザードの指示に従って、新規 PlanAhead プロジェクトを作成します。
2. [Design Source] ページで、次のいずれかをオンにします。
   a. [Specify RTL Sources] (RTL プロジェクトを作成する場合)
   b. [Specify synthesized (EDIF or NGC) netlist] (ネットリスト プロジェクトを作成する場合)
3. ウィザードを続行します。
4. 次のものを指定します。
   a. ソース
   b. ユーザー制約ファイル (UCF)
5. [Finish] をクリックします。

パーティションの設定

PlanAhead ソフトウェア でパーティションを設定するには、次の手順に従います。

1. Flow Navigator の [RTL Design] または [Netlist Design] をクリックして RTL デザインまたはネットリスト デザインをメモリに読み込みます。
2. [RTL Netlist] ビューまたは [Netlist] ビューをクリックしてデザイン階層を確認します。
3. パーティションするモジュール インスタンスを選択します。
4. 右クリックし、[Set Partition] をクリックします。
   パーティションは、パーティションに設定することを考慮して設計および合成したインスタンスにのみ設定してください。

PXML ファイルのインポート

Synplify コンパイル ポイント フローでは、Synplify で PXML ファイルを生成できます。詳細は、Symplify の sxml2pxml に関する資料を参照してください。

sxml2pxml ファイルの値を PlanAhead にインポートし、パーティションを定義できます。パーティションが既に定義されている場合は、オプションが淡色表示されます。

PXML ファイルをインポートするには、次の手順に従います。

1. Flow Navigator の [RTL Design] または [Netlist Design] をクリックして RTL デザインまたはネットリスト デザインをメモリに読み込みます。
2. [Netlist] ビューをクリックしてデザイン階層を確認します。
3. [Netlist] ビューを右クリックし、[Import Partition Settings] をクリックします。

PXML ファイルのエクスポート

ISE® Design Suite のコマンド ライン フローで使用する PXML を、PlanAhead ソフトウェアで作成できます。

PXML ファイルを作成するには、次の手順に従います。

1. RTL プロジェクトの場合は Flow Navigator の [RTL Design] をクリックして RTL デザインを、ネットリスト プロジェクトの場合は [Netlist Design] をクリックしてネットリスト デザインを読み込みます。
2. [RTL Netlist] ビュー (RTL プロジェクトの場合) または [Netlist] ビュー (ネットリスト プロジェクトの場合) をクリックします。
3. パーティションを設定するモジュール インスタンスを選択します。
4. 右クリックし、[Set Partition] をクリックします。
5. [Implement] ボタンの右側の矢印をクリックし、[Implementation Settings] をクリックします。
6. [Implementation Settings] ダイアログ ボックスで [Launch Options] フィールドの右側にあるボタンをクリックします。
7. [Launch Options] ダイアログ ボックスで [Generate scripts only] をオンにします。
8. [OK] をクリックします。
9. [Run] をクリックします。

PXML ファイルが、次のように生成されます。

- `<project_name>.runs/impl_1` ディレクトリに生成されます。
- ISE コマンド ライン フローを実行するディレクトリにコピーして使用できます。

階層デザイン フローの実行に関する詳細は、第 6 章「デザイン保持フロー」の「PlanAhead ソフトウェアでのデザイン保持フロー」を参照してください。

PlanAhead ソフトウェアの使用法の詳細は、付録 A「その他のリソース」にリストされている『PlanAhead ユーザー ガイド』(UG632) を参照してください。

BoundaryOpt 属性の設定

BoundaryOpt 属性を使用すると、パーティションの境界で一部の最適化を実行できます。この最適化は、インプリメンテーション結果にのみ影響します。XST ではサポートされていません。

パーティションに BoundaryOpt 属性を追加するには、次の手順に従います。

1. [Netlist] ビューでパーティションを選択します。
2. [Instance Properties] ビューで [Partition] タブをクリックします。
3. [Boundary Optimization] チェック ボックスをオンにします。
4. [Apply] をクリックします。

詳細は、第 2 章「設計に関する考慮事項」を参照してください。

ImportTag 属性の設定

ImportTag 属性を使用すると、1 つの階層で定義したパーティションを別の階層にインポートできます。この機能は、次でサポートされます。

- XST インクリメンタル フロー
- インプリメンテーション ツール

パーティションでの ImportTag 属性の設定

パーティションに ImportTag 属性を設定するには、次の手順に従います。

1. [(RTL) Netlist] ビューでパーティションを選択します。
2. [Instance Properties] ビューで [Partition] タブをクリックします。
3. [ImportTag] フィールドにパーティションがインプリメントされた階層名を入力します。
4. [Apply] をクリックします。

Tcl コンソールからの ImportTag 属性の設定

ImportTag 属性を Tcl コンソールから設定するには、次の Tcl コマンドを入力します。

```
set_property HD_IMPORT_TAG "<original_partition_name>" [get_cells"<current_instance_name>"]
```

ImportTag 属性を削除するには、同じコマンドを値を空にして実行します。

```
set_property HD_IMPORT_TAG "" [get_cells "<current_instance_name>"]
```

get_cells の値は、次のとおりです。

- パーティション名ではなくインスタンス名を指定します。
- 最上位モジュールまたはエンティティ名を含みます。

詳細は、第 1 章「パーティション」を参照してください。

パーティションのフロアプラン

パーティションをフロアプランするかどうかを判断するには、第 1 章「パーティション」の「パーティションを使用する状況の判断」を参照してください。

各パーティションに対して Pblock を作成するには、[Netlist] ビューでパーティションを 1 つずつ選択します。PlanAhead ソフトウェアで Pblock を作成すると、AREA_GROUP 制約が定義されます。

Pblock を作成するには、次の手順に従います。

1. パーティション インスタンスを右クリックします。
2. [Draw Pblock] をクリックします。
3. [Device] ビューで、各パーティションを配置する矩形を描きます。

PlanAhead ソフトウェアでモジュールをフロアプランする方法の詳細は、付録 A「その他のリソース」にリストされている『PlanAhead チュートリアル : デザイン解析とフロアプラン』(UG676) を参照してください。

パーティション デザインの合成

パーティションを含むデザインを合成するには、次の手順に従います。

1. RTL プロジェクトを作成します。
2. RTL デザインを表示してパーティションを定義します。
3. [Synthesize] をクリックします。

XST がデフォルトのインクリメンタル手法を使用して実行されます。各パーティションに個別の NGC ファイルが生成されます。

合成を実行する前に合成オプションを変更するには、次の手順に従います。

1. [Synthesize] ボタンの右側の矢印をクリックします。
2. [Synthesis Settings] をクリックします。
3. 設定を変更します。

パーティション デザインのインプリメント

パーティションを含むデザインをインプリメントするには、次のいずれかのセクションの手順に従います。

- 初心者ユーザー
- 上級ユーザー

初心者ユーザー

このセクションは、次のユーザー用です。

- 初心者ユーザー
- 最小限のセットアップを使用するユーザー

パーティションを含むデザインをインプリメントするには、次の手順に従います。

1. [Implement] をクリックします。
2. インプリメンテーション ツールがデフォルトのストラテジを使用して実行されます。

デフォルトのストラテジは、通常次のツールの ISE Design Suite でのデフォルト設定です。

- NGDBuild
- MAP
- PAR
- TRACE

インプリメンテーションを実行する前にインプリメンテーション オプションを変更するには、次の手順に従います。

1. [Implement] ボタンの右側の矢印をクリックします。
2. [Implementation Settings] をクリックします。
3. 設定を変更します。

上級ユーザー

このセクションは、上級ユーザー用です。

パーティションを含むデザインをインプリメントするには、次の手順に従います。

1. [Implement] ボタンの右側の矢印をクリックします。
2. [Create New Implementation Runs] をクリックします。

このオプションは、次のように使用できます。

- 異なるインプリメンテーション オプションを使用して、複数のインプリメンテーション実行を作成できます。
- SmartXplorer の実行に似ています。

インプリメンテーションの結果は、次の場所に示されます。

- [Compilation] ビュー
- ステータス バー
- [Window] → [Design Runs] をクリック

インプリメンテーションが完了したら、次の操作を実行します。

1. [Implemented Design] をクリックして結果を読み込みます。
2. パーティションが適切に定義されており、インプリメンテーション ツールで使用されているかどうかを確認します。

パーティションのプロモート

パーティションを合成またはインプリメントしたら、今後の実行でインポートできるようプロモートする必要があります。PlanAhead ソフトウェアでパーティションをプロモートするのは、コマンド ライン フローでパーティションをエクスポートするのと同じです。

アクティブな実行からパーティションをプロモートするには、Flow Navigator で [Promote Partitions] ボタンをクリックします。

実行結果をプロモートしないと、実行をリセットしたときに現在のパーティションが失われます。

結果をプロモートしたら、現在の実行をリセットして、再実行を必要なだけ繰り返します。プロモートされた結果は、パーティションが変更されていなければインポートされます。新しい結果をプロモートすると、現在プロモートされているデータが上書きされます。

複数の実行およびプロモートされたデータの管理については、付録 A「その他のリソース」にリストされている『PlanAhead ソフトウェア チュートリアル : デザイン保持の利用』(UG747) を参照してください。

パーティション属性の管理

次のパーティション属性を変更できます。

- [Action]
- [Import From]
- [Preservation]

[Action]

パーティションのステート (インプリメントするかインポートするか) を指定します。

- [Implement] : パーティションを合成またはインプリメントします。
- [Import] : 以前の実行をインポートします。

[Import From]

インポートするパーティションの場所を指定します。[Action] を [Implement] に設定した場合は、この値は [N/A] に設定します。

[Preservation]

パーティションをインポートする際の保持レベルを指定します。指定できる値は次のとおりです。

- [Routing] (デフォルト)
- [Placement]
- [Synthesis]

[Action] および [Import From] フィールドの管理

パーティションをプロポートすると、次のように設定されます。

- [Action] を [Import] に設定
- [Import From] をパーティションをプロモートした場所に設定

自動管理のオフ

この自動管理機能をオフにするには、[Promote Partitions] ダイアログ ボックスで [Automatically manage Partition action and import location] チェック ボックスをオフにします。

パーティションの再インプリメントが必要になった場合

PlanAhead ソフトウェアでは、パーティションの再インプリメントが必要になった場合、[Action] フィールドは自動的には [Implement] に戻されません。

プロモートされたパーティションは、次のような場合に再インプリメントが必要になります。

- ソース ファイル (HDL またはネットリスト) がアップデートされた場合
- パーティションに関連する物理制約が変更された場合

  これらはプロモートされたインプリメンテーション データにのみ影響し、プロモートされた合成データには影響しません。

再インプリメントが必要な状態になると、ソースがプロモートされているデータと一致せず、次のような NGDBuild エラーが生成されます。

```
ERROR:NgdBuild:1037 - The logic for imported Partition '/top/Express_Car'
   using previous implementation
   '../../project_1.promote/Ximpl_1/top_prev_built.ngd'
   has been modified.This situation can occur when the source for the
   Partition was modified but the Partition was not re-implemented
   and exported.
  You must re-implement and export the Partition before it can be imported into this design.
```

Preservation 属性の管理

PlanAhead ソフトウェアでは、Preservation 属性は管理されません。手動で管理する必要があります。

1. [Implement] ボタンの右側の矢印をクリックし、[Specify Partitions] をクリックします。
2. [Specify Partition] ダイアログ ボックスで [Preservation] の設定を変更します。

合成とインプリメンテーションのパーティション属性は独立しており、個別に設定します。合成とインプリメンテーションの両方で属性値を変更する必要がある場合もあります。

デザイン実行の管理

デザインは、次の場合に再インプリメントされます。

- デザインが変更されている。
- パーティションのステートが正しく設定されている。

推奨されるフロー

デザイン保持に推奨されるフローは、次のとおりです。

1. 1 つのデザイン実行 (impl_1) を保持します。
2. 必要に応じてプロモートおよび再インプリメントを繰り返します。

デザインの再インプリメンテーション

インプリメンテーションをプロモートした後、[Implement] をクリックしてデザインを再インプリメントします。

デザインを再インプリメントすると、次の処理が実行されます。

- 実行データがリセットされます。
- インプリメンテーション ツールが起動します。
- パーティション設定に従ってパーティションがインポートされます。

結果のプロモート

インプリメンテーションが問題なく終了したら、その結果をプロモートできます。以前にプロモートされていた結果は上書きされるので、デザイン パーティションに存在する promote ディレクトリは 1 つのみです。

複数のプロモート場所を作成し、複数の場所からパーティションをインポートするようにした方が有益な場合もあります。その場合は、正しい結果が得られるようプロモート場所およびインポート場所を適切に管理してください。

XILINX®

第 6 章

デザイン保持フロー

デザイン保持を使用すると、次のような利点があります。

- タイミング クロージャ段階でのインプリメンテーションの実行回数を削減できます。

  デザインの一部でタイミングが満たされたら、そのインプリメンテーション結果 (配置および配線) を次の実行で使用します。
- 完成し、タイミングを満たしている部分が、デザインのほかの部分を変更したときに影響を受けるのを防ぎます。
- パーティションを使用してモジュール　インスタンスの以前のインプリメンテーション結果を保持します。

パーティションを適切に使用すると、次のような利点があります。

- デザインの実行回数が削減されます。
- タイミングクロージャ段階の時間を短縮します。
- 変更されていないインスタンスを再検証する必要がなくなります。

インプリメンテーションのランタイム

デザインの一部を保持した場合、インプリメンテーションのランタイムは次の条件により異なります。

- インプリメントされるモジュール
- 保持されるモジュール

長いランタイム

インプリメントされるモジュールのタイミング要件が厳しい場合、ランタイムが長くなります。

短いランタイム

次のような場合は、ランタイムは短くなります。

- インプリメントされるモジュールのタイミングを簡単に満たすことができる。
- クリティカル パスが保持されるモジュールに含まれている。

コマンド ラインでのデザイン保持フロー

このセクションでは、コマンド ラインでのデザイン保持フローについて説明します。次の内容が含まれます。

- ユーザーによる管理
- コマンド ラインでのデザイン保持フローの手順

ユーザーによる管理

コマンドラインからデザイン保持フローを実行する際は、次をユーザーが管理する必要があります。

- どのモジュールがアップデートされたか。
- どのパーティションをインプリメントし、どのパーティションをインポートするのか。

次のガイドラインに従ってください。

- テキスト エディターまたは XML エディターで PXML ファイルを変更します。
- このフローを実行するのに特別なインプリメンテーション オプションは必要ありません。
- PXML ファイルをインプリメンテーションを実行するディレクトリに配置します。

PXML ファイルの作成および管理の詳細は、第 4 章「コマンド ラインでのパーティション フロー」を参照してください。

コマンド ラインでのデザイン保持フローの手順

コマンド ラインでのデザイン保持フローの手順は、次のとおりです。

1. デザイン サイクルの初期段階でデザインのパーティションを定義します。
2. デザイン コードを記述します。
3. インクリメンタル合成フローまたはボトムアップ合成フローを使用してデザインを合成します。

   サポートされる合成ツールおよびフローの詳細は、第 3 章「合成パーティションフロー」を参照してください。
4. PXML ファイルですべてのパーティションのステートを implement に設定してデザインをインプリメントします。
5. パーティションでタイミングが満たされたら、その結果を今後の実行用にエクスポートします。
6. バグの修正や機能の向上のためにデザインを変更したら、変更したパーティションを再インプリメントします。
7. ネットリスト レベルで変更されていないパーティションは、インポートして配置配線情報を保持します。
8. インポート　ディレクトリで指定されたパーティションに完全に一致しないパーティションをインポートしようとすると、NGDBuild ツールでエラー メッセージが表示されます。この場合、パーティションのステートを **`implement`** に変更する必要があります。
9. パーティション モジュールが完了し、タイミングが満たされていれば、デザインのほかの部分を変更している間、継続してインポートできます。

PlanAhead ソフトウェアでのデザイン保持フロー

このセクションでは、デザインを保持するために PlanAhead™ ソフトウェアを使用する方法を説明します。デザイン保持フローは、パーティション ベースのフローです。

詳細は、次を参照してください。

- 第 5 章「PlanAhead でのパーティション フロー」
- 『デザイン保持の利用』(UG747) (付録 A「その他のリソース」にリンクをリスト)

PlanAhead ソフトウェアでのデザイン保持フローの手順

PlanAhead ソフトウェアでのデザイン保持フローの手順は、次のとおりです。

1. 次のものを追加してレジスタ トランスファー レベル (RTL) PlanAhead プロジェクトを作成します。
   - RTL コード
   - ユーザー制約ファイル (UCF)
2. Flow Navigator で [RTL Design] をクリックして RTL デザインをエラボレートします。
3. 次のいずれかの方法でパーティションを定義します。
   - PlanAhead ソフトウェアを使用する。
   - 既存の PXML ファイルをインポートする。
4. すべてのパーティションのステートを **`implement`** に設定してデザインを合成します。

   各パーティションに個別の NGC ファイルが作成されます。
5. 必要に応じて次の制約を作成します。
   - 次のようなタイミング制約
     - PERIOD
     - OFFSET IN
     - OFFSET OUT
   - 次のようなフロアプラン制約
     - I/O ピンの LOC
     - AREA_GROUP
6. すべてのパーティションのステートを **`implement`** に設定してデザインをインプリメントします。
7. 適切なスタティックおよびダイナミック タイミング結果が得られたパーティションの合成結果およびインプリメンテーション結果をプロモートします。
8. プロモートしたパーティションをインポートし、合成およびインプリメンテーションを繰り返します。

ネットリスト デザイン保持フロー

このセクションでは、ネットリスト デザイン保持フローについて説明します。

ネットリスト デザイン保持フローの手順

ネットリスト デザイン保持フローの手順は、次のとおりです。

1. PlanAhead ソフトウェア外で、デザインのボトムアップ合成またはインクリメンタル合成を実行します。
2. 次のようにネットリスト ベースの PlanAhead ソフトウェア プロジェクトを作成します。
   - 手順 1 の合成結果を使用します。
   - 既存の UCF 制約ファイルを読み込みます。
3. 次のいずれかの方法でパーティションを定義します。
   - PlanAhead ソフトウェアを使用する。
   - 既存の PXML ファイルをインポートする。
4. 必要に応じて次の制約を作成します。
   - 次のようなタイミング制約
     - PERIOD
     - OFFSET IN
     - OFFSET OUT
   - 次のようなフロアプラン制約
     - I/O ピンの LOC
     - AREA_GROUP
5. すべてのパーティションのステートを **`implement`** に設定してデザインをインプリメントします。
6. 適切な結果が得られたパーティションのインプリメンテーション結果をプロモートします。

   適切な結果が得られたパーティションとは、適切なスタティックおよびダイナミック タイミング結果が得られたパーティションです。
7. PlanAhead ソフトウェア外で合成を繰り返します。
8. PlanAhead ソフトウェアを使用して、プロモートしたパーティションをインポートし、インプリメンテーションを繰り返します。

XILINX®

第 7 章

チーム デザイン フロー

チーム デザイン フローは、次のようなフローです。

- 階層デザイン手法です。
- パーティションを使用して、複雑な大型デザインを小型の論理ブロックに分割します。

これらのブロックは、個別に平行してインプリメントできます。この手法を使用すると、デザインのほかの部分を含むコンテキストで、各ブロックを個別に設計、インプリメント、および検証できます。

すべてのブロックが完了したら、その結果をインポートしてデザイン アセンブリを実行します。

チーム デザイン フローのチーム

チーム デザイン フローのチームは、次のメンバーで構成されます。

- 1 人のチーム リーダー (TL)
- 1 人または複数のチーム メンバー (TM)

チーム リーダー

チーム リーダーには、次のような設計者をお勧めします。

- ISE® Design Suite ソフトウェアの使用経験が豊富
- ターゲット アーキテクチャに関する知識がある

チーム リーダーの役割りは、次のとおりです。

- 最上位ロジックを設計、合成、インプリメンテーション、および検証します。
- タイミング制約およびフロアプラン制約含む最上位 UCF (ユーザー制約ファイル) を作成します。
- 最後にデザインをアセンブルします。

チーム メンバー

チーム メンバーの数に特に制限はありません。通常チーム メンバーの数は、デザインを分割するブロックの数になります。チーム メンバーは、割り当てられたパーティションを設計、合成、インプリメント、および検証します。

チーム デザイン フローの概要

チーム デザイン フローには、次のような特徴があります。

- パーティションを使用して各モジュールのタイミング クロージャおよび検証を個別に実行し、各チーム メンバー ブロックを最終デザインにインポートします。
- 各チーム メンバーが定期的に最新バージョンをパブリッシュし、チーム リーダーがそれらをアセンブルするプロセスを繰り返します。
- デザイン サイクルの初期段階でパーティション インターフェイスでのタイミングの問題を特定し、修正できます。

デザインが最終段階に近づき、各パーティションの配置が最適化されたら、最終アセンブリですべてのパーティションをインポートします。

最終アセンブリでタイミングが満たされない場合は、次の処理を実行します。

- 一部の'パーティションの保持レベルを緩和します。
- 一部のパーティションのステートを implement に設定します。

既にアセンブルされたデザインも、同様の方法でアップデートできます。再インプリメントが必要なのはアップデートされたパーティションのみで、トップ パーティションを含むその他のパーティションはインポートできます。

チーム デザイン フローのダイアグラム

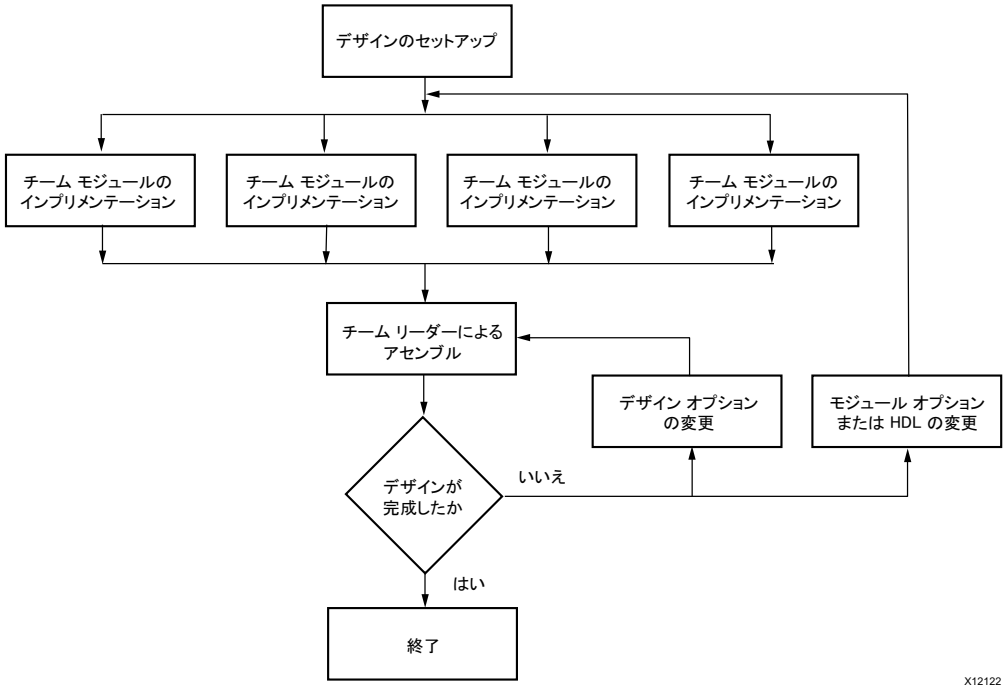

図 7-1 : チーム デザイン フローのダイアグラム

チーム デザイン フローのセットアップ

チーム リーダーがチーム デザイン フローをセットアップします。

チーム デザイン フローのセットアップには、次の手順があります。

- 階層の定義
- RTL の作成
- プロジェクトの設定
- 初期合成
- 制約の定義
- 初期インプリメンテーション
- プロモートまたはエクスポート
- プロジェクトの配布

チーム デザイン フローのセットアップのダイアグラム

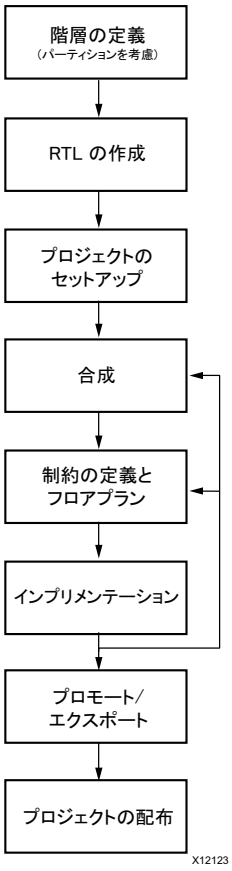

図 7-2 : チーム デザイン フローのセットアップ

階層の定義

適切な階層には、次のような特徴があります。

- 最適化およびパックの制限による影響を最小限に抑えます。
- デザインが正しくインプリメンテーションできるかどうかに重要な役割を果たします。
- チーム メンバーのパーティションを定義するため、階層レベルを追加してはっきりとした境界を定義する必要がある場合があります。

チーム リーダーがデザインのプランを立て、パーティションを定義します。

RTL の作成

初期設定では、次を含む最小限のデザインが必要です。

- すべての I/O ポートを含む最上位モジュール
- すべてのチーム メンバー ブロックのインスタンシエーションを含む正しい階層
- 可能な限り多くのグローバル ロジック (クロックおよびリセット)
- トップ パーティションの IP コア (オプション)

各チーム メンバー ブロックは、ブラック ボックスから完全なレジスタ トランスファー レベル (RTL) までを使用できます。ポートの幅と方向を定義するため、モジュールまたはコンポーネント宣言が必要です。

プロジェクトの設定

ISE Design Suite および PlanAhead™ ソフトウェアの両方で、チーム デザイン フローがサポートされます。どちらでも、ある程度の初期設定が必要になります。次を決定する必要があります。

- コマンド ライン フローまたは PlanAhead ソフトウェア フロー
- 合成ツール
- 合成のタイプ
- 合成およびインプリメンテーションのオプション
- ソース制御

チーム デザイン フローでは、ソースを柔軟に管理できます。各チーム メンバーがそれぞれのソース ファイルを管理することをお勧めします。これには、次のようにします。

- バージョン管理システムを使用します。
- 各チーム メンバーがローカルにサンドボックスを用意し、必要に応じてアップデートします。

コマンド ライン フローおよび PlanAhead ソフトウェア フローの両方で、この方法がサポートされています。PlanAhead ソフトウェアを使用している場合、ソースをプロジェクトにコピーできるので、すべてのチーム メンバーが共通ソースをローカルにコピーして使用できます。

このようにすると、ほかのチーム メンバーに影響を与えずにソースをアップデートできます。すべてのチーム メンバーが同じソースを使用する場合、ソースを変更するとすべてのチーム メンバーに影響します。ソースの変更により、エクスポートされたインプリメンテーション データが最新ではなくなります。上記の推奨事項に従うと、共通のリポジトリをアップデートする前に変更をテストおよび検証できます。

初期合成

初期合成では、すべてのモジュールが正しく定義されていることを確認します。

チーム メンバーのモジュールには、次の状態のものを使用できます。

- 完全な RTL または部分的な RTL
- ブラック ボックスのモジュール定義

チーム デザイン フローでは各ブロックを個別に作業するので、次の要件があります。

- フラット トップダウン合成フローは使用できません。
- 次のいずれかの合成フローを使用する必要があります。
  - ボトムアップ合成フロー
  - トップダウン インクリメンタル合成フロー

ボトムアップ合成では、チーム メンバーのパーティションはトップ パーティションのコンテキスト外で合成されるので、チーム メンバーはトップ パーティション用に合成済みネットリストのみが必要です。インプリメンテーションは、コンテキスト内で実行されます。

チーム デザイン フローは、ISE Design Suite でサポートされる Xilinx Synthesis Technology (XST) インクリメンタル フローに基づいています。このフローでは各チーム メンバーのパーティションはトップ パーティションのコンテキスト内で合成されるので、各チーム メンバーは次が必要です。

- 最上位 HDL コード
- すべてのチーム メンバーのパーティションで結果を保持するため、合成中にトップ パーティションをインポートできます。

詳細は、第 3 章「合成パーティションフロー」を参照してください。

制約の定義

チーム リーダーが、次を含む最上位ユーザー制約ファイル (UCF) を作成します。

- 次のような I/O ロジック制約および属性
  - I/O および LOC 配置制約
  - IOSTANDARD タイミング制約
  - OFFSET IN タイミング制約
  - OFFSET OUT タイミング制約
- 次のようなグローバル クロックのタイミング制約および配置制約
  - BUFG、PLL、および MMCM コンポーネントの LOC 制約
  - PERIOD 制約
- AREA_GROUP 制約を使用した各チーム メンバーのパーティションの配置

チーム リーダーは、各パーティションが必要なリソースにアクセスできるようクロック リソースおよびグローバル リソースを管理する必要があるので、チーム リーダーはターゲット アーキテクチャに関する知識が必要です。

初期インプリメンテーション

チーム メンバーのインプリメンテーションも含め、すべてのインプリメンテーションは最上位ロジックのコンテキスト内で実行します。初期インプリメンテーションでは、デザインはほとんど作成されていないので、有益な配置およびタイミング結果が得られない場合があります。

次のことを確認できます。

- すべてのモジュールが正しく解決するか。
- パーティションが正しく定義されているか。
- すべてのタイミング制約と物理制約の構文が正しいか。

ブラック ボックスとして定義されているパーティションは、合成とインプリメンテーションでサポートされており、特別なオプションや属性は必要ありません。チーム メンバーのパーティションがすべてブラック ボックスであっても、パーティション フローを使用するので、これらのブロックをインプリメントすることは可能です。NGDBuild ツールでパーティションがブラック ボックスとしてインプリメントされることを示す警告メッセージが表示されます。

プロモートまたはエクスポート

合成およびインプリメンテーションの結果は、どの段階でもプロモートまたはエクスポートできます。

初期のデザイン セットアップ段階では、トップ パーティションまたはチーム メンバー モジュールの有益なインプリメンテーション データはほとんどありませんが、最上位ロジックにパーティションされた完了した IP コアがあることもあります。

各チーム メンバー パーティションはトップ パーティションのコンテキスト内でインプリメントされるので、IP コアをエクスポートし、各チーム メンバーがインプリメンテーションを実行する際にインポートできます。

XST インクリメンタル フローでは、チーム メンバーが最上位合成結果をインポートし、すべてのチーム メンバー パーティションが一定の最上位を使用できるようにします。ボトムアップ合成フローでは、各チーム メンバーの合成はトップ パーティションのコンテキスト外で実行されるので、これは不要です。

プロジェクトの配布

次の時点でチーム リーダーが各チーム メンバーにプロジェクト ファイルを配布します。

- パーティションが定義された。
- 制約が作成された。
- デザインが合成およびインプリメントされた。

コマンド ラインでのプロジェクトの配布

コマンド ラインでは、プロジェクトの配布に次のものが含まれます。

- 合成プロジェクト
- HDL コード
- UCF ファイル

PlanAhead ソフトウェアでのプロジェクトの配布

PlanAhead ソフトウェアでは、プロジェクトの配布に各チーム メンバー用のチーム リーダー プロジェクトのコピーが含まれます。

チーム リーダーは、[File] → [Save Project As] をクリックして各チーム メンバー用にコピーを作成します。これにより、すべてのチーム メンバーが同じデータに基づいて設計を開始できます。

チーム メンバーの役割り

各チーム メンバーの役割りは、それぞれのモジュールを開発して検証し、タイミング クロージャを達成することです。このプロセスは、デザイン サイクル中繰り返されます。

各チーム メンバーの手順は、次のとおりです。

- モジュールの開発およびアップデート (チーム メンバー)
- モジュールの合成 (チーム メンバー)
- インプリメンテーション (チーム メンバー)
- エクスポートまたはプロモート (チーム メンバー)
- インポート (チーム メンバー)

チーム メンバーの開発フローのダイアグラム

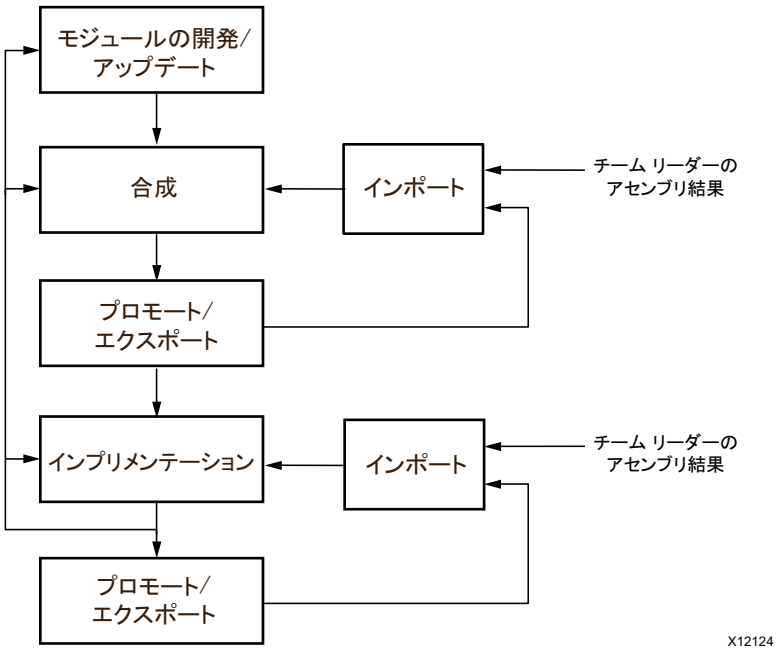

図 7-3 : チーム メンバーの開発フロー

モジュールの開発およびアップデート (チーム メンバー)

モジュールの開発およびアップデートでは、HDL コードを作成および変更します。

チーム メンバーが次の操作を実行します。

- HDL コードを作成および変更します。
- 次の場合に変更をチーム リーダーに送付します。
  - パーティションのインターフェイスを変更した。
  - 新しい機能をインプリメントした。
  - チーム メンバー パーティションに変更がある。

チーム リーダーは、次の操作を実行します。

- 最上位をアップデートします。
- アップデートした最上位を各チーム メンバーに配布します。

モジュールの合成 (チーム メンバー)

各チーム メンバーのモジュールは、次のように合成できます。

- ボトムアップ フローを使用してトップ パーティションからは独立して合成

  チーム リーダーがインプリメンテーション用に各チーム メンバーに最上位ネットリストを配布します。

- インクリメンタル フローを使用してトップ パーティションのコンテキストで合成
  - トップ パーティションのコンテキストでモジュールを合成するため、各チーム メンバーに最上位 HDL コードが必要です。
  - 各チーム メンバーがチーム リーダーの最新の実行結果をインポートし、すべてのメンバーで同じ実行結果が使用されるようにします。

インプリメンテーション (チーム メンバー)

チーム メンバーは、それぞれのブロックをトップ パーティションのコンテキスト内で実行します。

次の事項に従うことをお勧めします。

- ほとんどの場合、トップ パーティションを implement に設定します。
- ほかのチーム メンバーのブロックは、ブラック ボックスとしてインポートまたはインプリメントします。
- ほかのチーム メンバーのパーティションは、すべてのパーティションのインターフェイス タイミングを満たしたチーム リーダーのアセンブリ実行のものをインポートします。

ほかのチーム メンバーのパーティションをブラック ボックスとしてインプリメントしている場合のタイミング クロージャについては、第 1 章「パーティション」の「ブラック ボックスの使用法」を参照してください。

タイミング クロージャの達成方法

チーム メンバーは、次を含む使用可能なすべての方法を使用してタイミング クロージャを達成します。

- 次のような物理制約を作成
  - AREA_GROUP
  - LOC
- 次のような特殊なタイミング制約を作成
  - 複数サイクル パス
  - フォルス パス
- ソフトウェア オプションを使用
- SmartXplorer を使用

制約の送付

チーム メンバー パーティションをインポートする場合は、すべての制約をチーム リーダーに送付する必要はありませんが、チーム リーダーがチーム メンバー パーティションをインプリメントする場合を考慮していつでも送付できる状態にしておく必要があります。

チーム リーダーの操作

チーム メンバーがブロックの新しいバージョンを送付したら、チーム リーダーは次の操作を実行します。

- デザインの最新バージョンと共にタイミングを検証します。
- 新しい結果をすべてのチーム メンバーにエクスポートします。

インプリメントまたはインポート

各チーム メンバーは、デザインが完成に近づくまでは、最上位をインポートするのではなくインプリメントしてインプリメンテーションを実行することをお勧めします。デザインが完成に近づいてきたら、トップ パーティションをインポートするかインプリメントするかを決定できます。

エクスポートまたはプロモート (チーム メンバー)

最終アセンブリで予測されない結果が得られるのを回避するため、デザイン プロセス全体を通して、各チーム メンバーは結果をチーム リーダーにエクスポート (コマンド ライン) またはプロモート (PlanAhead ソフトウェア) することをお勧めします。このプロセスにより、発生する可能性のある問題を開発中に特定し、修正できます。

インポート (チーム メンバー)

各チーム メンバーは、合成およびインプリメンテーション時にほかのパーティションをインポートまたはインプリメントできます。この操作には柔軟性がありますが、各チーム メンバーが次のように操作することをお勧めします。

- インクリメンタル合成を使用する場合は、合成でトップ パーティションをインポートします。
- デザインが完成に近づき、アセンブリ実行からのエクスポートされた結果が最終デザインの配置配線を正しく表すようになるまでは、インプリメンテーションでトップ パーティションをインプリメントします。
- インクリメンタル合成を使用する場合は、合成でほかのチーム メンバーのブロックをインポートします。
- インプリメンテーションでほかのチーム　メンバーのブロックをインポートするか、ブラックボックスとします。

注記：各チーム メンバーは、ほかのチーム メンバーのブロックをそれぞれのメンバーから直接インポートするのではなく、チーム リーダーがプロモートしたデータをインポートすることをお勧めします。これにより、すべてのチーム メンバーがデザインの同じバージョンに基づいて作業することを確実にできます。

チーム リーダーの役割り

チーム リーダーは、アセンブリ フローを管理します。アセンブリ フローでは、デザインを完全にアセンブリし、次の状態にします。

- 配置配線を完了する。
- タイミングを満たす。

アセンブリは、次の要素によって簡単な場合と困難な場合があります。

- デザインの密度および速度
- デザイン フローを通してチーム メンバーどれだけ緊密に作業したか

アセンブリは、反復作業です。チーム メンバーは、新しい結果が得られ、チーム リーダーおよびほかのチーム メンバーと共有できるようになったら、チーム リーダーのアセンブリ実行用にデータをエクスポートします。

アセンブリ実行の結果によって、チーム リーダーは次のいずれかの操作を実行します。

- チーム メンバーと協力して問題を解決します。
- 結果に問題がない場合は、すべてのチーム メンバーにエクスポートします。

アセンブリ フローのダイアグラム

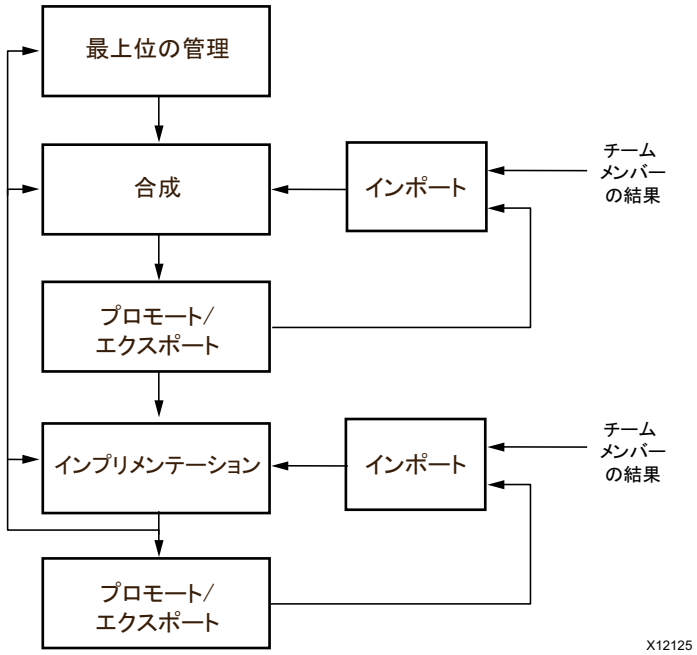

図 7-4 : アセンブリ フロー

最上位の管理 (チーム リーダー)

最上位は、チーム リーダーが開発して管理します。

トップ パーティションにどれだけのロジックがあるかによって、各チーム メンバーがそれぞれブロックを開発している間、チーム リーダーもデザイン プロセス全体を通してトップ パーティションの開発を続ける必要がある場合もあります。

トップ パーティションに含まれているのが主に I/O とクロックである場合、初期デザイン セットアップの後に必要な開発はそれほどありません。

合成 (チーム リーダー)

インクリメンタル XST フローを使用している場合は、チーム リーダーは次を実行できます。

- 各チーム メンバーから最新の RTL コードを取得し、デザイン全体を合成します。
- 各チーム メンバーがエクスポートした最新の合成結果をインポートします。

デザイン全体を再合成する必要がなければ、最新の NGC ファイルをインポートするだけで十分です。これにより、チーム リーダーとチーム メンバーの間での一貫性を保つことができます。

インプリメンテーション (チーム リーダー)

アセンブリ段階のインプリメンテーションでは、各チーム メンバーからの最新の結果を使用します。チーム リーダーは、次の操作を実行します。

- トップ パーティションのタイミングの問題を解決します。
- チーム メンバー パーティション間のタイミングの問題を特定します。
- 制約ファイルを管理します。

チーム メンバーのユーザー制約ファイル (UCF)

各チーム メンバーが、最上位 UCF のほかに UCF を使用していることがあります。チーム メンバーの UCF には、そのチーム メンバー ブロックに特定のタイミング制約および配置制約が含まれています。

これらの制約をチーム リーダーのインプリメンテーションに含めるには、チーム リーダーは次のいずれかの操作を実行します。

- ファイルを手動で結合します。
- 複数の UCF ファイルを PlanAhead プロジェクトに追加します。
- 複数の UCF ファイルをコマンド ライン (NGDBuild ツール) で指定します。

複数の UCF ファイルがある場合は、重複する制約がないようにする必要があります。同じロジックに対して異なる値の 2 つの制約が設定されている場合、インプリメンテーション ツールで最後に読み込まれた制約が使用されます。この競合により、結果が予測されないものになることがあります。

チーム メンバー ブロックのインポートまたはインプリメント

インプリメンテーションでは、チーム リーダーは各チーム メンバー ブロックをインプリメントまたはインポートします。

- チーム メンバー ブロックをインプリメントすると、次のようになります。
  - トップ パーティションをより柔軟に配置できます。
  - デザイン サイクルの初期段階では、タイミングの問題を特定するために必要な場合があります。
- 各チーム メンバー ブロックをインポートするのは、次のような状態になってからにします。
  - ブロックが完成に近づいている。
  - パーティション インターフェイスのタイミングが満たされている。

最終アセンブリ

すべてのチーム メンバー ブロックが完成したら、最終アセンブリを実行します。最終アセンブリは、次のように実行します。

- 各チーム メンバーのモジュールをインポートします。
- トップ パーティションは、タイミング結果によってインプリメントまたはインポートします。
- チーム メンバー ブロックの保持レベルは緩和できます。

  保持レベルを緩和すると、インポートしたチーム メンバー ブロックの配置配線を調整できます。
- 次のような場合は、個々のパーティションをインプリメントできます。
  - 1 つのチーム メンバー ブロックに少しの変更が加えられている。
  - アセンブリ実行でタイミングが満たされない。

プロモートまたはエクスポート (チーム リーダー)

各チーム メンバーがパブリッシュするブロックが完成に近づいてくると、インプリメンテーション結果もより統合されたものに近づきます。チーム リーダーは、最新の合成およびインプリメンテーション結果をエクスポートまたはプロモートし、各チーム メンバーに配布します。

注記：各チーム メンバーは、ほかのチーム メンバーのブロックをそれぞれのメンバーから直接インポートするのではなく、チーム リーダーがプロモートしたデータをインポートすることをお勧めします。これにより、すべてのチーム メンバーがデザインの同じバージョンに基づいて作業することを確実にできます。

ほとんどの場合、チーム リーダーはインプリメンテーション データのみをエクスポートしますが、次のようにすることもできます。

- 合成結果をプロモートして、すべてのチーム メンバーがほかのチーム メンバーのブロックの最新ネットリストを使用できるようにします。
- 各チーム メンバー ブロックをインポートする際に最上位の合成に対して反復実行します。

インポートまたはインプリメント (チーム リーダー)

チーム リーダーは、次の段階で各チーム メンバー ブロックをインポートするかインプリメントするかを決定します。

- 合成
- インプリメンテーション

合成でのチーム メンバー ブロックのインポートまたはインプリメント

チーム リーダーは、合成で各チーム メンバー ブロックをインポートするかインプリメントするかを決定します (インクリメンタル合成を使用する場合)。チーム リーダーが次を実行することをお勧めします。

- チーム メンバーの結果をインポートします。
- XST の **read_cores** オプションを使用して、各ブロックの最新の NGC ファイルを読み込みます。

インプリメンテーションでのチーム メンバー ブロックのインポートまたはインプリメント

チーム リーダーは、インプリメンテーションで各チーム メンバー ブロックをインポートするかインプリメントするかを決定します。可能な限り、チーム メンバー ブロックをインポートすることをお勧めします。

最上位またはほかのチーム メンバーのブロックへのインターフェイス タイミングが満たされない場合は、一部のチーム メンバー ブロックをインプリメントする必要がある場合があります。

保持レベルの変更 (チーム リーダー)

チーム リーダーは、インポートするパーティションの保持レベルを次のいずれかの設定に変更できます。

- placement
- synthesis

保持レベルの placement への変更

チーム リーダーは、配線競合が発生している場合にインポートするパーティションの保持レベルを placement に変更できます。保持レベルを placement に変更すると、配線ツールで配線を変更できるようになります。

保持レベルの synthesis への変更

チーム リーダーは、次の場合にパーティションの保持レベルを synthesis に変更できます。

- タイミングが満たされていない。
- 配置競合が発生している。

保持レベルを synthesis に変更すると、インプリメンテーション ツールで配置配線を変更できるようになります。チーム デザイン フローのエリア グループ制約の要件に従っていれば、配置競合は発生しません。

パーティション ステートの変更

チーム リーダーは、次のような場合にパーティションのステートを import から implement に変更します。

- 保持レベルを緩和してもデザインが完了しない。
- 既にアセンブルされたデザインに少しの変更が加えられている。

すべてのパーティション デザインの設計に関する推奨事項

次の推奨事項は、チーム デザイン フローを含むすべてのパーティション デザインに適用されます。

- パーティション境界の両側で信号にレジスタを付けます。
- ロジックをデザインのほかの部分から独立した機能を持つグループに分離します。
- 一緒に配置する必要のあるロジックを 1 つのパーティションに含めます。

チーム デザイン フロー特定の設計に関する推奨事項

次の推奨事項は、チーム デザイン フロー特有のものです。

- チーム メンバー ブロックを特定の領域にフロアプラン
- パイプライン レジスタ段の追加
- ブラック ボックス モジュール
- 最上位デザイン

チーム メンバー ブロックを特定の領域にフロアプラン

最上位チーム リーダー ロジック以外の各チーム メンバー ブロックは、AREA_GROUP 制約を使用してデバイスの特定の領域にフロアプランする必要があります。この制約には、次のような機能があります。

- ロジックの配置を制御します。
- アセンブリ中の配置競合を回避します。

AREA_GROUP 制約のオーバーラップ

チーム メンバー間でAREA_GROUP 制約をオーバーラップさせることはできません。

ブロックの配線の制御

AREA_GROUP 制約を使用して、ブロックの配線を制御します。次のような場合、アセンブリ中に配線競合が発生する可能性があります。

- デザインで配線が密集している
- チーム メンバー ブロックが別のチーム メンバー ブロックの隣または近くにある

各チーム メンバー ブロックの配線を制御すると、次のような利点があります。

- 配線競合を削除または削減できる
- アセンブリの信頼性が向上する

配線は各チーム メンバーの実行で制御する必要があるので、配線制御はデザイン サイクルの初期段階で決定する必要があります。

その他の必要な制約

AREA_GROUP の配線を制御するには、次の 2 つの制約が必要です。

- CONTAINED=ROUTE

  AREA_GROUP に含まれる配線リソースのみを使用するよう指示します。

- PRIVATE=NONE

  最上位からのロジックをチーム メンバーの AREA_GROUP 内に配線できるようにします。

  注記：これは AREA_GROUP の通常の動作ですが、CONTAINED=ROUTE を使用する場合はこの制約を設定する必要があります。

これらの制約は、配線を制御するチーム メンバー ブロックに適用されます。

制約の構文例

次に示す制約の構文例では、**cpuEngine** というインスタンスを使用します。

```
INST "cpuEngine" AREA_GROUP = "pblock_cpuEngine";
AREA_GROUP "pblock_cpuEngine" RANGE=SLICE_X64Y60:SLICE_X105Y119;
AREA_GROUP "pblock_cpuEngine" CONTAINED=ROUTE;
AREA_GROUP "pblock_cpuEngine" PRIVATE=NONE;
```

CONTAINED=ROUTE の動作

次に、CONTAINED=ROUTE の動作に関する重要な情報を示します。

パーティションされたインスタンスに属するコンポーネント

パーティションされたインスタンスには、RANGE 制約で明示的に指定されたコンポーネントのみが含まれます。

配線を含むパス

配線は、ドライバーとロードの両方がチーム メンバー パーティションに含まれるパスにのみ含まれます。

パスは論理的にチーム メンバー ブロックに含まれているが、ドライバーまたはロードのどちらかが物理的にはパーティションに含まれない場合もあります。例として、デバイスの中央に配置されている次のようなグローバル ロジックが挙げられます。

- BUFG
- MMCM
- SYSMON
- ICAP

チーム メンバー パーティションに物理的に含まれていないパス

チーム メンバー パーティションに物理的に含まれていないパスには、チーム メンバー エリア外の配線リソースを使用できます。

- これらのパスはチーム メンバー パーティションに含まれていないため、アセンブリ中に配線競合が発生する可能性があります。
- このようなパスを回避または最小限に抑えるには、クロック ロジックおよび特殊コンポーネントを最上位パーティションに配置します。

チーム メンバー パーティションに物理的に含まれるパス

配線は、チーム メンバー パーティションに物理的に含まれるパスに含まれます。これらのパスでは、チーム メンバー パーティションに含まれる配線リソースのみを使用できます。

パーティションの AREA_GROUP の配線が過度に使用されている場合は、パーティションで使用されないコンポーネントに RANGE 制約を追加する必要があることがあります。RANGE 制約を追加すると、次の両方の条件を満たす場合に有益です。

- チーム メンバー パーティションでブロック RAM や DSP コンポーネントなどの一部のコンポーネントを使用しない場合
- スライスの RANGE がこれらの未使用のコンポーネントを完全に含む

これらの未使用のコンポーネントに RANGE 制約を追加すると、次のような利点があります。

- チーム メンバー ブロックに配線リソースを含めることができます。
- パーティションで使用可能な配線が増加します。
- 配線、タイミング、またはその両方が向上する可能性があります。

パイプライン レジスタ段の追加

可能な場合、1 つのパーティションから別のパーティションへのタイミング クリティカル パスに、トップ パーティション内でパイプライン レジスタ段を追加します。タイミング クリティカル パスには、次のものがあります。

- チーム リーダーからチーム メンバー
- チーム メンバーからチーム メンバー

チーム デザイン フローのエリア グループ要件のため、このようにすると配置が制限されたロジック (チーム メンバー ブロックなど) へのタイミングを満たしやすくなります。

ブラック ボックス モジュール

ブラック ボックス モジュールがサポートされます。チーム メンバー モジュールがまだ存在していない場合や、タイミングまたはランタイムのためにブラック ボックスを使用するのが有益な場合は、合成およびインプリメンテーションの両方でブラック ボックスにしておくことができます。詳細は、第 1 章「パーティション」の「ブラック ボックスの使用法」を参照してください。

最上位デザイン

最上位デザインは、次のようにします。

- すべての I/O バッファーを含めます。
- 含めるロジックは、少量にします。

「少量」とは、デザインによって異なります。通常は、トップ パーティションに含まれるロジックが多くなると、最終アセンブリがより複雑になります。チーム リーダーが管理するトップ パーティションに子パーティションを設定すると、アセンブリでインプリメントするロジックの量を最小限に抑えることができます。

コマンド ライン フロー

ISE Design Suite コマンド ライン フローには、特別なオプションやツールは不要です。パーティションのすべての情報は、PXML ファイルに保存されます。

SmartXplorer

チーム リーダーおよびチーム メンバーは、SmartXplorer を使用してデザインに適切なインプリメンテーション オプションを判断できます。詳細は、第 5 章「PlanAhead でのパーティション フロー」の「パーティションを含むデザインでの SmartXplorer の使用」を参照してください。

PXML ファイル

チーム デザイン フローをコマンド ラインから実行する場合は、チーム リーダーおよび各チーム メンバーがそれぞれの PXML を管理する必要があります。PXML ファイルの名前は `xpartition.pxml` にする必要があります。

PXML ファイルの例は、次を参照してください。

- 「PXML ファイルの例 (チーム リーダー)」
- 「PXML ファイルの例 (チーム メンバー)」

ディレクトリ構造 (コマンド ライン フロー)

チーム デザイン プロジェクトでは、ディレクトリ構造に特に要件はありません。通常、ソース ファイルは Perforce や CVS などのバージョン管理システムを使用して管理されます。各チーム メンバーは、リポジトリからローカル サンドボックスにチェック アウトします。

次のディレクトリ構造の例を開始点として使用できます。

チーム リーダーのディレクトリ構造 (コマンド ライン フロー)

- TL
  - Synth

    トップ パーティションおよびチーム メンバー パーティションの合成結果

    メモ : ボトムアップ合成プロジェクトまたはインクリメンタル合成プロジェクトの場合
  - HDL
    - チーム メンバー ブロックの VHDL または Verilog ソース ファイル
    - まだ存在していないチーム メンバー パーティションのブラック ボックス モジュールの定義
  - UCF
    - 最上位 UCF ファイル
    - チーム メンバー特定の UCF ファイル
  - Impl
    - 最上位インプリメンテーション結果
    - 各チーム メンバー デザインの最新バージョン
    - `xpartition.pxml`

- XImpl
  - エクスポートまたはプロモートされた最新のインプリメンテーション結果

    チーム メンバーは、このディレクトリからトップ パーティションおよびほかのチーム メンバーのブロックをインポートします。この場所にアクセスできないチーム メンバーがいる場合は、適切なエクスポート場所を選択してください。
- Assemble
  - 各チーム メンバー ブロックをインポートするデザインの最終インプリメンテーション
  - トップ パーティションは、タイミング結果によってインプリメントまたはインポートします。

チーム メンバーのディレクトリ構造 (コマンド ライン フロー)

- TM
  - Synth
    - チーム メンバー ブロックの合成結果 (ボトムアップ合成プロジェクトの場合)
    - トップ パーティションおよびチーム メンバー ブロックの結果 (インクリメンタル合成プロジェクトの場合)

      チーム メンバー ブロックに含まれる HDL が少ない場合は、プロキシ HDL ファイルの合成結果です。このプロキシ HDL ファイルには、ほかのパーティションへのインターフェイス タイミングを満たすため、すべての入力および出力にレジスタを付けます。
  - HDL
    - デザイン全体の VHDL または Verilog ソース ファイル
    - トップ パーティションの VHDL または Verilog ソース ファイル (インクリメンタル合成フローの場合)
    - まだ存在していないチーム メンバー パーティションのブラック ボックス モジュールの定義
  - UCF
    - 最上位 UCF ファイル
    - LOC や AREA_GROUP など、チーム メンバー パーティション特定の物理制約を含むチーム メンバーの UCF ファイル
  - Impl
    - チーム メンバー ブロックのインプリメンテーション結果 (ほかのチーム メンバー ブロックはインポート、トップ パーティションはインプリメントまたはインポート)
    - `xpartition.pxml`
  - XImpl
    - エクスポートされた最新のチーム メンバー インプリメンテーション結果

      チーム リーダーがこれをインポートし、アセンブリ実行に使用できます。

      チーム リーダーがこの場所にアクセスできない場合は、適切なエクスポート場所を選択してください。

チーム リーダーの役割り (コマンド ライン フロー)

チーム リーダーは、最新のトップ パーティションおよび各チーム メンバー ブロックを使用してデザイン全体をインプリメントします。このフローのすべてのインプリメンテーションと同様、パーティションは各チーム メンバー ブロックとトップ パーティションに設定されます。下位パーティションをデザインに追加した場合、トップもパーティションにする必要があります。

PXML ファイルの例 (チーム リーダー)

```
<?xml version="1.0"?>
<Project Name="impl_1" FileVersion="1.2" ProjectVersion="2.0">
  <Partition Name="/top" State="implement" ImportLocation="NONE">
    <Partition Name="/top/U0" State="implement" ImportLocation="NONE" Preserve="routing">
    </Partition>
    <Partition Name="/top/U1" State="implement" ImportLocation="NONE" Preserve="routing">
    </Partition>
    <Partition Name="/top/U2" State="import" ImportLocation="../../TM3/Ximpl"
Preserve="routing">
    </Partition>
  </Partition>
```

チーム メンバー ブロックが完成に近づいてきたら、最終アセンブリで各チーム メンバー パーティションのステートを import に設定します。**ImportLocation** は、チーム メンバー パーティションがエクスポートされているディレクトリに設定します。インプリメンテーションが正常に完了したら、チーム リーダーはその結果をチーム メンバーにエクスポートします。

チーム メンバーはトップ パーティションをインポートまたはインプリメントできますが、ほかのチーム メンバーのブロックはチーム リーダーがエクスポートした結果を常にインポートする必要があります。

チーム メンバーの役割り (コマンド ライン フロー)

各チーム メンバーは、それぞれのデザインを、トップ パーティションのコンテキスト内でほかのチーム メンバーのパーティションをインポートしてインプリメントします。各チーム メンバーは、通常はトップ パーティションをインプリメントします。その方が、チーム メンバー ブロックを柔軟にインプリメントできます。

デザインが完成に近づき、チーム メンバーがそれぞれのブロックを調整する段階になったら、トップ パーティションをインポートするとインターフェイス タイミングを保持できます。

デザインを正常にインプリメントできたら、チーム メンバーは次の操作を実行します。

- インプリメンテーション結果をチーム リーダーにエクスポートします。
- ブロックの最新のソース (ネットリストおよび HDL) をチーム リーダーに渡します。

これにより、ソースとエクスポートされたパーティション データが一致し、NGDBuild ツールでロジック変更検出エラーが誤って表示されるのを回避できます。

デザインの最新バージョンでインプリメンテーションを実行したら、チーム リーダーはその結果をチーム メンバーにエクスポートします。

注記 : 各チーム メンバーは、ほかのチーム メンバーのブロックをそれぞれのメンバーから直接インポートするのではなく、チーム リーダーがプロモートしたデータをインポートすることをお勧めします。これにより、すべてのチーム メンバーがデザインの同じバージョンに基づいて作業することを確実にできます。

PXML ファイルの例 (チーム メンバー)

```
<?xml version="1.0"?>
<Project Name="impl_1" FileVersion="1.2" ProjectVersion="2.0">
  <Partition Name="/top" State="import" ImportLocation="../../TL/Ximpl ">
    <Partition Name="/top/U0" State="implement" ImportLocation="NONE" Preserve="routing">
    </Partition>
    <Partition Name="/top/U1" State="import" ImportLocation="../../TM2/Ximpl"
Preserve="routing">
    </Partition>
    <Partition Name="/top/U2" State="import" ImportLocation="../../TM3/Ximpl"
Preserve="routing">
    </Partition>
  </Partition>
</Project>
```

PlanAhead ソフトウェア フロー

PlanAhead ソフトウェアの環境外で合成を実行し、ネットリスト ベースの PlanAhead ソフトウェア プロジェクトを使用してチーム デザイン フローをインプリメントすることは可能ですが、このセクションでは RTL プロジェクトを XST インクリメンタル フローで実行することを前提として説明します。特別な PlanAhead ソフトウェア設定やライセンスは必要ありません。詳細は、第 5 章「PlanAhead でのパーティション フロー」を参照してください。

チーム リーダーの役割り (PlanAhead ソフトウェア フロー)

PlanAhead ソフトウェア フローでは、チーム リーダーは次の手順に従います。

1. 初期 RTL PlanAhead ソフトウェア プロジェクトを作成します。
   - プロジェクトに、デザインのこの段階でのすべてのソース ファイルが含まれる場合があります。
   - プロジェクトには、チーム メンバー モジュールのブラック ボックスをインスタンシエートする最上位 HDL ファイルを含める必要があります。
   - プロジェクトは、この章で前述したソース管理の推奨事項に従っている必要があります。
2. RTL デザインでチーム メンバー ブロックにパーティションを定義します。
   - パーティションを定義するには、HDL をエラボレートする必要があります。
   - HDL にエラーがあるためにエラボレートできない場合は、パーティションは定義できません。
3. 合成を実行します。
4. 問題のなかった結果をプロモートします。
5. 必要に応じて次のようなグローバル制約を追加または作成します。
   - 次のようなタイミング制約
     - PERIOD
     - OFFSET
     - FROM:TO

- 次のような物理制約
  - チーム メンバー ブロックの AREA_GROUP 制約
  - クロック ロジックおよび I/O の LOC 制約

  これは、フロアプランと呼ばれます。

6. 次を実行してこの時点でのデザインを検証します。
   - PlanAhead ソフトウェアのデザイン ルール チェック (DRC) を実行
   - インプリメンテーションを実行

   この時点では有益なインプリメンテーション結果は得られないかもしれませんが、デザインを検証することでインプリメンテーション ツールの論理的および物理的 DRC チェックを使用してデザインおよび制約を確認できます。
7. [Project Save As] コマンドを使用して、各チーム メンバー用に 1 つずつプロジェクトを作成します。

   各チーム メンバーがそれぞれのブロックを開発、インプリメント、プロモートしたら、この手順を繰り返してすべてのチーム メンバーのプロジェクトがチーム リーダーのプロジェクトと一致しているようにします。次の図を参照してください。

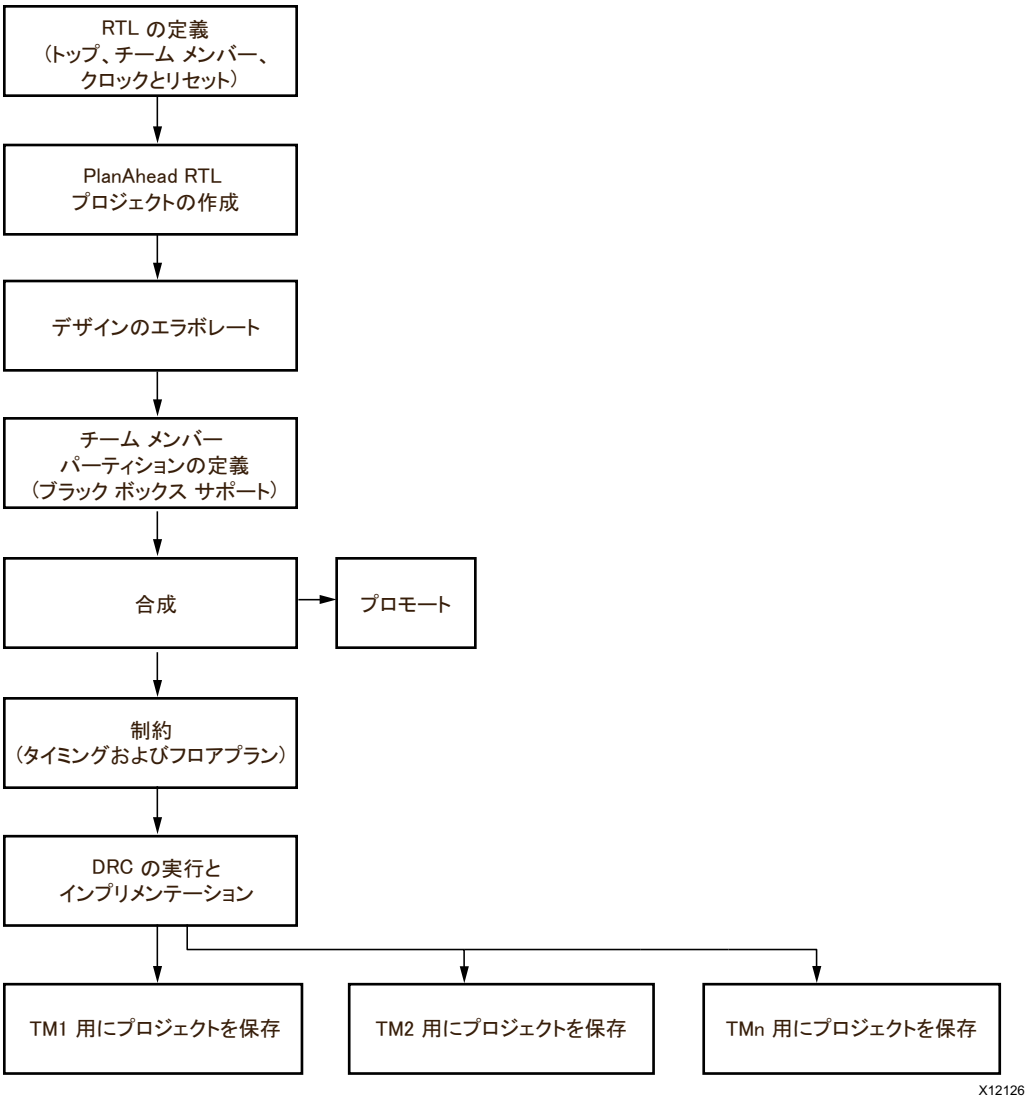

図 7-5 : チーム リーダーの初期操作 (PlanAhead フロー)

インターフェイス タイミング

タイミングを満たし、タイミングを保持するためにパーティション間のインターフェイス タイミングを指定するのは、困難な場合があります。通常、1 つのパーティションをインプリメントしてほかのパーティションをインポートしますが、ロジックを最適な場所に配置できないこともあります。

この問題を解決するには、使用するフローによって次の 2 つの方法があります。

- チーム メンバー パーティションをブラック ボックスとして定義する

  1 つのチーム メンバー パーティションのインターフェイス ロジックに制約を設定し、ほかのチーム メンバー パーティションはブラック ボックスとして定義します。

- チーム メンバー パーティションをインターフェイス ロジックとして定義する

  すべてのインターフェイス ロジックが存在している状態で (ブラック ボックスなし)、チーム メンバー パーティション間のインターフェイス タイミングを制約し、満たし、保持します。

チーム メンバー パーティションをブラック ボックスとして定義する

次の場合、インターフェイス タイミングをパーティション ピンへの制約を使用して定義できます。

- チーム メンバーが自分のロジックのみをトップ パーティションと共にインプリメントしている。
- ほかのチーム メンバー パーティションはブラック ボックスとして定義されている。

パーティションのピン タイミング制約

チーム リーダーは最初、パーティション ピンのタイミング制約を作成します。ブラック ボックス モジュールのプロキシ ロジックを制約する例は、第 1 章「パーティション」の「ブラック ボックスの使用法」を参照してください。チーム リーダーはその後、次の操作を実行します。

イベント	操作
• パーティションのピン タイミング制約の定義	1. チーム メンバー パーティションをブラック ボックスとして定義 2. デザインをインプリメント
• インプリメンテーション完了 • インターフェイスのタイミングが満たされる	1. ブラック ボックス パーティションをプロモート 2. パーティションのピン ロケーションをエクスポート

チーム メンバーは、自分のブロックをインプリメントする際にこのプロモートされたデータと適切なタイミング制約を使用して、次を達成します。

- インターフェイス ロジックをパーティションの AREA_GROUP の適切なエッジの近くに配置します。
- アセンブリでタイミングを満たします。

プロキシ ピンのロケーションをチーム リーダーの初期インプリメンテーションから変更する必要がある場合は、**pr2ucf** を使用して制約を抽出します。

```
pr2ucf top_routed.ncd -o partition_pins.ucf
```

これらの制約は、次のように変更します。

- 手動で変更できます。
- PlanAhead ソフトウェアでは変更できません。

変更されたロケーションは、次のいずれかの方法で各チーム メンバーに配布します。

- デザイン フローに UCF ファイルを追加します。
- 制約を最上位 UCF に統合します。

パーティション ピンのロケーションは UCF 制約で固定するので、各チーム メンバーのブラック ボックス モジュールは、インポートするのではなくインプリメントする必要があります。

この時点で、チーム リーダーは次の操作を実行できます。

- 変更されたパーティション ピンのロケーションを使用してブラック ボックスを再インプリメントします。
- 各チーム メンバー用に結果を再度エクスポート (またはプロモート) します。

1 つのモジュールをインプリメントするときにほかのチーム メンバー パーティションをブラック ボックスとして定義すると、メモリ使用量を削減し、ランタイムを短縮できますが、次の理由によりインターフェイス タイミングが不正確になる可能性があります。

- クロック スキュー、ファンアウト、または適切なロジック レベル数が考慮されていない。
- インターフェイス タイミングの問題がアセンブリ中に発生する可能性がある。

このフローを使用してメモリの使用量を削減する方法の詳細は、第 1 章「パーティション」の「大型デザインのメモリ使用量の管理」を参照してください。

チーム メンバー パーティションをインターフェイス ロジックとして定義する

デザイン サイクルの初期段階では、ほとんどの場合インターフェイス タイミングは無視できます。各チーム メンバーは自分のブロックをインプリメントする際に、ほかのチーム メンバーのブロックをインポートするか、ブラック ボックスとして定義します。チーム メンバー ブロックのインターフェイス ロジックが定義されたら、コードをチーム リーダーに送付します。

チーム リーダーの操作

チーム リーダーは、次の操作を実行します。

- すべてのパーティションをインプリメントします。
- インターフェイスのタイミング問題を特定し、解決します。
- 結果をプロモートし、すべてのチーム メンバーが使用できるようにします。

チーム メンバーの操作

チーム メンバー A は自分のブロックを開発およびインプリメントし、チーム リーダーがプロモートした結果からチーム メンバー B のブロックをインポートします。

チーム メンバー B も、同様にチーム メンバー A のブロックをインポートします。パーティション インターフェイスのタイミングは変更されません。

フローの利点

このフローの利点は、次のとおりです。

- インターフェイス タイミングは 100% 正確です。
- インターフェイスのタイミング問題は、変更が加えられているフローの初期段階で特定できます。
- 特別なタイミング制約または配置制約を作成する必要はありません。ほとんどの場合、通常の PERIOD 制約があります。

チーム メンバーの役割り (PlanAhead ソフトウェア フロー)

各チーム メンバーに、チーム リーダー プロジェクトと同一の PlanAhead ソフトウェア プロジェクトが配布されます。

チーム メンバーは、次の操作を実行します。

1. ソースを開発し、プロジェクトに追加します。
2. 合成、制約の定義、インプリメンテーションを実行し、結果をプロモートします。

プロジェクトが分岐し始めます。チーム メンバーは、チーム リーダーのプロジェクトと定期的に同期させることをお勧めします。

チーム メンバーの結果が、ほかのチーム メンバーおよびチーム リーダーが使用できる状態になったら、合成およびインプリメンテーションの結果をチーム リーダーがアクセスできる場所にプロモートします。

チーム リーダーは、次の操作を実行します。

1. デザインの最新バージョンを使用して、合成またはインプリメンテーション、あるいはその両方を実行します。
2. 結果をプロモートしてチーム メンバーが使用できるようにします。

注記 : 各チーム メンバーは、ほかのチーム メンバーのブロックをそれぞれのメンバーから直接インポートするのではなく、チーム リーダーがプロモートしたデータをインポートすることをお勧めします。これにより、すべてのチーム メンバーがデザインの同じバージョンに基づいて作業することを確実にできます。

XILINX®

第 8 章

パーティションのデバッグ

この章では、パーティションのデバッグについて説明します。次の内容が含まれます。

- インプリメンテーション エラー
- インポートするパーティションの属性値について
- BitGen の DRC エラー
- ChipScope のサポート

インプリメンテーション エラー

インプリメンテーション エラーには、次のようなものがあります。

- パーティションをインポートできない
- デザインがタイミングを満たさない
- デザインを配置できない
- デザインを配線できない
- パーティションによりスライスの使用量が増加する

パーティションをインポートできない

次のような理由でパーティションをインポートできない場合があります。

- ソース ネットリストの変更
- ソフトウェアの変更
- PXML ファイルがインポート ディレクトリにコピーされない

ソース ネットリストの変更

次の時点の間でソース ネットリストが変更されていると、パーティションをインポートできない場合があります。

1. パーティションがエクスポートされた時点
2. パーティションがインポートされた時点

これが、最もよくあるインポートできない理由です。

ソフトウェアの変更

次の時点の間でソフトウェアが変更されていると、パーティションをインポートできない場合があります。

1. パーティションがエクスポートされた時点
2. パーティションがインポートされた時点

以前のソフトウェア バージョンからのパーティションのインポートは、機能するとは限りません。リリース間で属性値や DRC ルールが変更されている可能性があります。これらの変更に対応するため、インポートされたパーティションを変更したり、再インプリメントしたりする必要がある場合があります。これは、特にテスト中の新しいアーキテクチャに当てはまります。

たとえば、前のバージョンのソフトウェアで問題なく実行されていたデザインで、MMCM に対して次のような DRC エラーが発生する場合があります。

```
ERROR:PhysDesignRules - The calculated CLKIN1_PERIOD frequency/DIVCLK_DIVIDE value of
100.000000 for MMCM_ADV instance clocks/dcm200/DCM_ADV is less than the allowed lower limit
of 135.0 MHz, when BANDWIDTH is HIGH or OPTIMIZED.Change BANDWIDTH to LOW to correct this
problem.
```

PXML ファイルがインポート ディレクトリにコピーされない

PXML ファイルがインポート ディレクトリにコピーされていないと、パーティションをインポートできないことがあります。

この場合、次のようなエラー メッセージが表示されます。

```
ERROR:HierarchicalDesignC:154 - Import project "./import" is not an XPartition project.
ERROR:HierarchicalDesignC:143 - Error importing Partition "/top/module".
ERROR:HierarchicalDesignC:142 - Error found in XPartition file data.
```

デザインがタイミングを満たさない

デザインのパーティションされている部分でタイミングが満たされない場合、インポートされたパーティションの保持レベルを synthesis に緩和すると、インポートされたパーティションの配置および配線を必要に応じて変更でき、ツールでより柔軟な処理が可能となるため、タイミングが満たされる可能性が高くなります。

タイミングが満たされたら、次のようにします。

- パーティションをエクスポートします。
- 保持レベルを必要に応じて placement または routing に変更します。

保持レベルを緩和してもデザインのタイミングが満たされない場合は、次のようにします。

- 1 つまたは複数のパーティションのステートを import から implement に変更します。
- クリティカルパスを含むパーティションを削除します。

標準フラット フローのタイミング クロージャ手法を適用できます。

デザインを配置できない

デザインのパーティション数を多くし、インポートされるロジックの割合を高くすると、インプリメントされるロジックで使用可能なリソースが少なくなり、デザインを配置できないことがあります。

この場合、配置で次のようなメッセージが表示されます。

```
WARNING:Place:1178 - 4 sites were unavailable during the placement
phase because they were already used by routing within preserved
Partitions.If the Placer is not able to find a successful solution, the
following suggestions might help the Placer find a solution.
1) To allow the Placer to use these sites, back off the preservation
level on one or more of the following Partitions.The following
Partitions contain unavailable sites:
        Partition /top/Control_Module:3 (preserve=routing)
        Partition /top/Express_Car:1 (preserve=routing)
If the problem persists, then careful floorplanning can minimize the
number of sites occupied by routing.
```

これらのサイトを使用できないのは、インポートされた配線に含まれるピンによりコンポーネントが使用不可能になっているからです。配線ツールでインポートされた配線を移動することができれば、これらのコンポーネントにロジックを配置できます。これには、使用不可能なサイトの多いパーティションで保持レベルを placement または synthesis に変更します。これでも十分でない場合は、パーティションを再インプリメントする必要があります。

パーティションのステートを implement に変更

トップ パーティションのステートを implement に設定すると、配置ツールで有効な配置を見つけるのに十分なリソースが使用可能になる可能性があります。これでも十分でない場合は、そのほかのパーティションのステートも implement に変更してみてください。

PlanAhead™ ソフトウェアを使用して、どのパーティションのステートを implement に変更したらよいのかを判断します。

1. 結果ビューアーに配置配線情報を読み込みます。
2. 各パーティションをハイライトして FPGA のどこに位置しているかを確認します。

パーティション数の削減

パーティションの数が多すぎると、デバイスの使用率に悪影響を与える場合があります。デザインのクリティカルでない部分のモジュールからパーティションを削除すると、その他のロジックで使用できるリソースが増え、ツールでタイミングを満たすためのソリューションをより柔軟に見つけることができます。

フロアプラン

適切なフロアプランを使用すると、配置を見つけるのに役立つ場合があります。すべてのロジックをチップ全体のどこにでも配置可能な場合、配置ツールでインプリメントされる新しいロジック用のリソースを見つけることができない場合があります。

フロアプランでは、各パーティションを特定の部分に制限します。AREA_GROUP RANGE を追加または変更すると、エクスポートされたパーティション データのアップデートが必要となるので、次の実行でパーティションのステートを implement に設定する必要があります。

デザインを配線できない

インポートしたパーティションを使用してデザインを配線できない場合は、次のようにします。

1. 保持レベルを placement に変更します。
2. PAR を再実行します。

MAP を再実行する必要はありません。

フロアプラン

フロアプランが、配置の問題を解決するのに役立つのと同様の理由で、配線の問題を解決するのに役立つ場合があります。

保持レベルの placement への変更

保持レベルを変更したり、フロアプランを適用しても配線が完了しない場合は、配置されたデザインを解析すると、ステートを変更する必要のあるパーティションを判断できます。

PlanAhead™ ソフトウェアを使用して、どのパーティションのステート placement に変更したらよいのかを判断します。

1. 結果ビューアーに配置配線情報を読み込みます。
2. 各パーティションをハイライトして FPGA のどこに位置しているかを確認します。

パーティションによりスライスの使用量が増加する

パーティションを使用すると、最適化およびパッキングの制限により、スライスの使用量が多少増加することが予測されますが、これは通常数パーセントか、無視できる程度です。

スライスの使用量が大幅に増加する場合

LUT とフリップフロップの比率が大きく異なっていると、スライスの使用量が大幅に増加することがあります。

- パーティションが使用されていないデザインでは、モジュール A のコンポーネントの一部をモジュール B のコンポーネントと結合できます。
- パーティションを使用したデザインでは、パッキングが制限され、パーティション A のコンポーネントをパーティション B のコンポーネントと共にパックすることはできません。このため、LUT とフリップフロップのバランスが取られていないロジックで空のスライスが使用されることになり、スライスの使用量が増加します。

エリア グループおよびパーティション サマリ

デザインでこの問題が発生しているかどうかを確認するには、マップ レポート (.mrp) でエリア グループおよびパーティション サマリを参照してください。

マップ レポートのパーティション リソース サマリの例

```
Partition "/top/moduleA":
  State=implement
  Slice Logic Utilization:
    Number of Slice Registers:            4,318 (4,318)
    Number of Slice LUTs:                 4,337 (4,337)
      Number used as logic:               3,717 (3,717)
      Number used as Memory:                620 (620)
  Slice Logic Distribution:
    Number of occupied Slices:            1,750 (1,750)
    Number of LUT Flip Flop pairs used:   5,249 (5,249)
      Number with an unused Flip Flop:    931 out of  5,249   17%
      Number with an unused LUT:          1,508 out of  5,249   28%
      Number of fully used LUT-FF pairs:2,810 out of  5,249   53%
  Number of Block RAM/FIFOs:              3 (3)
  Number of BUFDS:                        1 (1)
  Number of BUFR:                         1 (1)
  Number of GTX_DUAL:                     2 (2)
  Number of PLL_ADV:                      1 (1)
```

リソース使用率の数値

マップ レポートには、使用量に対して次の 2 つの値が示されています。

- 1 つ目の値は個々のパーティションのリソース使用量を示します。
- かっこに含まれている 2 つ目の値はそのパーティションと子パーティションすべての値です。

この情報はマップ レポートのパーティション リソース サマリ セクションの最初に記述されており、見逃されがちです。次に例を示します。

```
Partition Resource Summary:
---------------------------
Resources are reported for each Partition followed in parenthesis by
resources for the Partition plus all of its descendents.
```

インポートするパーティションの属性値について

注意：インポートされた MMCM で属性を変更した場合、ソフトウェアでエラーは検出されず、以前の値が使用されます。変更は失われます。

MMCM (または同様のコンポーネント) の属性を変更しており、そのインスタンスをインポートすると、次のように処理されます。

- 以前の値がインポートされ、新しい値が上書きされます。
- エラーまたは警告メッセージは表示されません。

インポートされたパーティションの属性を HDL または UCF 制約で変更した場合は、パーティションを必ずインプリメントしてください。インプリメントしないと、新しい値は失われます。

BitGen の DRC エラー

デザインで PAR ツールの実行が問題なく完了しても、BitGen で次のような DRC エラーが発生することがあります。

```
ERROR:PhysDesignRules:10 - The network <signal_OBUF> is completely
unrouted
```

これは、パーティションの駆動されていない出力が、親パーティションのロジックに接続されていることが原因です。フラットフローのように親パーティションのロジックをマップで最適化できないため、部分的に配線された信号が最終デザインに含まれることになり、BitGen で DRC エラーが発生します。

この問題は、パーティションに **BoundaryOpt** 属性を使用すると修正されることがあります。この属性を使用すると、マップ ツールで未接続のポートを削除できるようになります。

BoundaryOpt 属性には制限があるため、問題を解決するのに HDL レベルでの修正が必要となる場合もあります。いずれの場合でも、修正されたパーティションを再インプリメントする必要があります。

ChipScope のサポート

ChipScope™ ツールでパーティションはサポートされています。

パーティションで ChipScope コアを変更した場合は、パーティション ステートを implement に変更する必要があります。

XILINX®

付録 A

その他のリソース

ザイリンクス リソース

- 『ISE® Design Suite : インストールおよびライセンス ガイド』(UG798) : http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/iil.pdf
- 『ISE Design Suite 13 : リリース ノート ガイド』(UG631) : http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/irn.pdf
- ザイリンクス用語集 : http://japan.xilinx.com/company/terms.htm
- ザイリンクス サポート : http://japan.xilinx.com/support/
- デバイス ユーザー ガイド : http://japan.xilinx.com/support/documentation/user_guides.htm
- データシート : http://japan.xilinx.com/support/documentation/data_sheets.htm

ISE 資料

- ISE 資料 : http://japan.xilinx.com/support/documentation/dt_ise13-3.htm
  - 『コマンド ライン ツール ユーザー ガイド』(UG628) : http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/devref.pdf
  - 『制約ガイド』(UG625) : http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/cgd.pdf
  - 『タイミング クロージャ ユーザー ガイド』(UG612) : http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/ug612.pdf
  - 『合成/シミュレーション デザイン ガイド』(UG626) : http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/sim.pdf
  - 『XST ユーザー ガイド (Virtex®-4、Virtex-5、Spartan®-3 および CPLD デバイス用)』(UG627) : http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/xst.pdf
  - 『XST ユーザー ガイド (Virtex-6、Spartan-6、および 7 シリーズ デバイス用)』(UG687) : http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/xst_v6s6.pdf

PlanAhead 資料

- PlanAhead 資料 :
  http://japan.xilinx.com/support/documentation/dt_planahead_planahead13-3.htm
- 手法ガイド :
  - 『フロアプラン手法ガイド』(UG633) :
    http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/Floorplanning_Methodology_Guide.pdf
  - 『ピン配置手法ガイド』(UG792) :
    http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/ug792_pinplan.pdf
  - 『PlanAhead ユーザー ガイド』(UG632) :
    http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/PlanAhead_UserGuide.pdf
  - 『PlanAhead Tcl コマンド リファレンス ガイド』(UG789) :
    http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/ug789_pa_tcl_commands.pdf
- PlanAhead チュートリアル :
  http://japan.xilinx.com/support/documentation/dt_planahead_planahead13-3_tutorials.htm
  - 『デザイン解析およびフロアプラン』(UG676) :
    http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/PlanAhead_Tutorial_Design_Analysis_Floorplan.pdf
  - 『I/O ピンの配置』(UG674) :
    http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/PlanAhead_Tutorial_IO_Pin_Planning.pdf
  - 『デザイン保持の利用』(UG747) :
    http://japan.xilinx.com/support/documentation/sw_manuals/xilinx13_3/PlanAhead_Tutorial_Design_Preservation.pdf