SYNOPSYS®

テスト・モデルを使用した階層スキャン合成メソドロジ

大規模/高性能スキャン合成テクノロジ・バックグラウンダ

2001年10月

目次

・概要
・はじめに
・テスト・モデルを使用した階層スキャン合成
・DFT Compilerでテスト・モデルを使用した階層スキャン合成の設計フロー
・インターフェイス・ロジック・モデル(ILM)とテスト・モデル
・結論

概要

システムオンチップ(SoC)テクノロジに移行し、設計の複雑さが数百万ゲート規模に急増する中、タイミング、面積、消費電力、およびテスト設計の迅速な収束が不可欠になっています。設計者は、シノプシスの論理合成および物理合成テクノロジを用いて今日の複雑な設計をインプリメントしています。このような大規模な設計のテスト合成をチップレベルで処理するには、一定レベルの抽象化が必要となります。抽象化により、システム/チップのインテグレーションを行う設計者は、テスト合成をインプリメントし、設計のやり直しを減らして、タイミングとテスト容易化設計(DFT)の収束を達成できます。論理および物理合成フェーズで抽出したタイミングおよび配置情報とともに、DFT情報をテスト・モデル形式で抽象化することにより、設計者は設計サイクルの初期段階でテスト構造の設計についての基本的な決定を行うことができます。また、この抽象化により、数百万ゲート規模の設計時にメモリ使用量を削減し、実行速度が飛躍的に向上した階層テスト・インプリメンテーションが可能になります。

はじめに

今日の複雑な設計では、DFTはテスタビリティの目標を達成し、効果的な製造テストを行うために、設計プロセスに不可欠な要素となりつつあります。すでにDFT Compilerは、スキャンと論理合成および物理合成を緊密に統合するワンパス・スキャン合成フローをサポートしています。この合成フローを使用することにより、シームレスなスキャン挿入を行い、テスト・ロジックのすべての設計制約条件を満たすことができます。また、設計の物理情報を考慮してスキャン・チェーン構築による配線混雑を最小限に抑えることができます。

複雑な設計では、ハードウェア記述言語(HDL)と合成ツール・コマンド・スクリプトを使用してシステム仕様を設計する場合、システム設計者は階層アプローチを優先手法として用います。次に、そのシステムを制約条件付きモジュールに分割します。その際、初期のフロアプラン情報を利用することにより、現実的な制約条件を設定できます。論理合成では、2種類のモジュール制約条件が存在します。「設計ルールの制約条件」は、機能設計が満たさなければならないテクノロジ固有の制約条件を反映したものです。テクノロジ固有の制約条件は最大ネット・ファンアウトなどを含み、通常ターゲット・テクノロジ・ライブラリと配置情報によって決定されます。「最適化の制約条件」は、デザインの機能を妨げない設計目標と条件です。例としては、最大遅延や面積などがあります。これらの制約条件付きサブモジュールそれぞれにおいてテスタビリティが処理され、続いてタイミング、面積、消費電力、および物理制約条件と共にすべてのテスタビリティ条件を満たすために、テスト構造がチップ・レベルで構築されます。システム設計者は、サインオフ・モジュールを統合してシステム記述を作成します。システム・ネットリストは、フロアプラン、配置、および配線のために物理設計ツールに引き渡されます。レイアウト後にタイミング情報が抽出され、合成済みのシステム記述にバックアノテートされます。制約条件がチェックされ、違反が存在する場合は、デザインが再最適化されます。この結果、最適化と設計ルールの制約条件を満たすシステムが完成します。多くの高品質SoCインプリメンテーションには、スキャンDFTが不可欠となっています。困難な機能目標を抱える設計者にとっては、設計サイクルの初期段階でDFTを考慮し、スキャンを合成手法に統合することが重要となります。

シノプシスは、スキャンを階層合成手法に統合するにあたって、2種類の手法を推奨しています。最初のカテゴリは、トップダウン手法です(次頁図1)。トップダウン手法では、システム統合が完成するまでスキャン挿入が延期されます。モジュール設計者は、デザインがプリ・スキャン設計ルール制約条件を満たしていることを確認します。またシステム設計者も、統合済みのシステムがプリ・スキャン設計ルール制約条件を満たしていることを確認します。次に、適切なスキャン制約条件を設定し、スキャン挿入、デザイン・ルール・チェック、および再最適化を行います。レイアウト時には、物理的なセルの位置などの物理設計情報を利用して、レイアウトに基づくスキャン・チェーン・リオーダリングを行うことができます。システム・インテグレータは、スキャン後の設計ルール・チェックを行って、リオーダリングによるスキャン設計ルール違反が発生していないことを確認する必要があります。トップダウン手法によっては、物理設計情報が利用可能になるまでスキャン・チェーン配線が延期されることもあります。シノプシスは、Physical CompilerにDFT Compilerを統合してワンパス・スキャン・オーダリングを実現しました。この統合は合成と配置を行い、配線混雑度を改善したDFTとタイミング両方の収束を実現するために配置情報に基づいたスキャン・チェーンの構築を実現します。

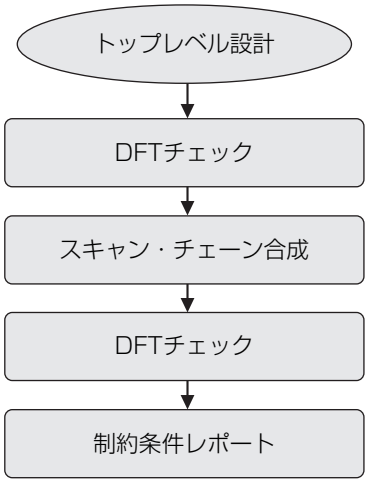

図1: トップダウン・スキャン合成フロー

シノプシスが推奨する2番目の階層手法は、大規模設計のスキャン合成に適したボトムアップ手法です(図2)。システム設計者は、設計およびスキャン制約条件を設定し、それらを個々のモジュールの設計者に渡します。モジュール設計者は、スキャン合成を実行し、モジュールが機能的なタイミング制約条件を満たしていることを確認します。システム設計者は、統合されたシステムがプリ・スキャン設計ルール・チェックを満たしていることを確認し、トップレベルでスキャン合成を実行します。システム設計者は、レイアウト後にPhysical Compilerを用いた環境で物理情報を利用したスキャン・チェーン・オーダリングを行います。

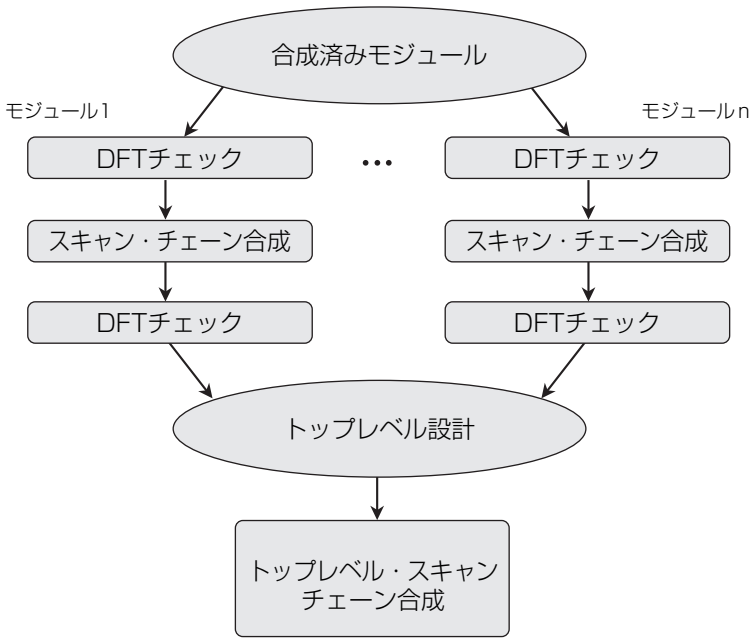

図2: 従来の階層ボトムアップ・スキャン合成フロー

ボトムアップ・スキャン挿入には、以下のような利点があります。

・設計フローの早い段階でのスキャン設計ルール違反の検出
・時間がかかるトップレベル再最適化のリスクの低減
・スキャン・モジュールの並行開発とサインオフの容易化

トップダウン・スキャン挿入には簡素化というメリットがありますが、大規模な数百万ゲート規模の設計では合成ツールの容量と処理時間の限界があります。しかし、回路規模が数百万ゲートに達し、サブモジュールの相対サイズも増大する中、ボトムアップ・アプローチを採用したとしても、合成ツールはすでにトップレベル・スキャン合成の容量の限界に達しています。このため、サブモジュール・レベルで一定レベルの抽象化が必要となります。これらのモジュールをトップレベルで統合することにより、大規模なデザインの性能と容量が向上します。シノプシスは、テスト・モデルを使用した階層スキャン合成手法を発表しました。この手法では、IEEE 1450.6 Core Test Language (CTL)規格を採用して、次世代の複雑な設計とSoCに対応したスキャン合成を実行します。

テスト・モデルを使用した階層スキャン合成

業界全体の取り組みの一環として、IEEEでは、SoC設計のテスト時にプラグ・アンド・プレイを実現できるようにテスト・テクノロジの要素の標準化を進めています。CTLは、システム統合時のすべてのテスト・ニーズを満たしつつ、テスト・インフラストラクチャとテスト・パターンの再利用に必要な情報の記述を目的として提唱されている規格です。

シノプシスは、スキャンとその他のテスト関連情報をテスト・モデル(ctl-db)として抽象化するために他社に先駆けてCTL規格を活用しました。このテスト・モデルは、DFT Compilerを用いたスキャン合成時に作成されますが、ユーザからは見えません(図3)。次にユーザは、テスト関連の情報のみを含むテスト・モデルと一般的なボトムアップ・フローで通常行われるフルゲート・ネットリストを出力します。トップレベルでは、トップレベル・ネットリストとともにテスト・モデルだけがメモリに読み込まれ、サブモジュールのすべてのゲートレベル情報を読み込むことなくスキャン構築が実行されます。このように、テスト・モデルを読み込んでトップレベル・スキャン設計ルール・チェックとともにスキャン構築を実行することで、大規模設計の処理容量と性能を飛躍的に向上させます。

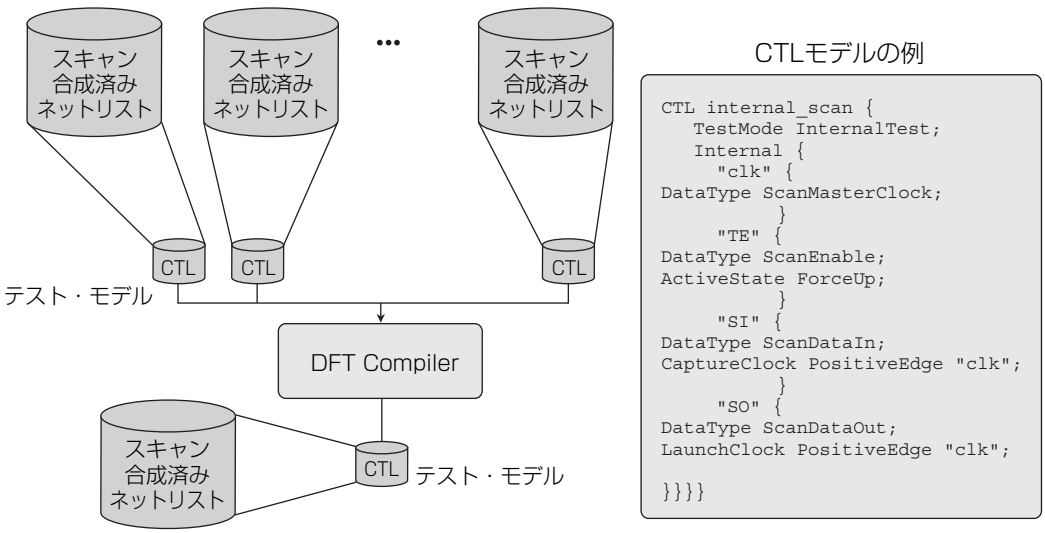

図3: テスト・モデルを使用した階層スキャン合成

DFT Compilerでテスト・モデルを使用した階層スキャン合成の設計フロー

図4は、今日のASICおよびSoCで50万から100万ゲート規模のサブモジュールの一般的なボトムアップ・スキャン合成フローを示したものです。この同じボトムアップ・フローは、DFT Compilerでスキャン・チェーン合成を行った後に自動的に作成されメモリに格納されるテスト・モデルの出力に用いるために拡張されます。各サブモジュールのテスト情報はテスト・モデルで表現され、データベース・フォーマットまたはHDLフォーマットのフル・ゲートレベル・ネットリストとは別に、シノプシスのインターナル・データベース(ctl-db)フォーマットで保存できます。このテスト・モデルは、タイミングおよびその他の設計属性を含むフル・ゲートレベル・データベースに比べて非常にコンパクトです。

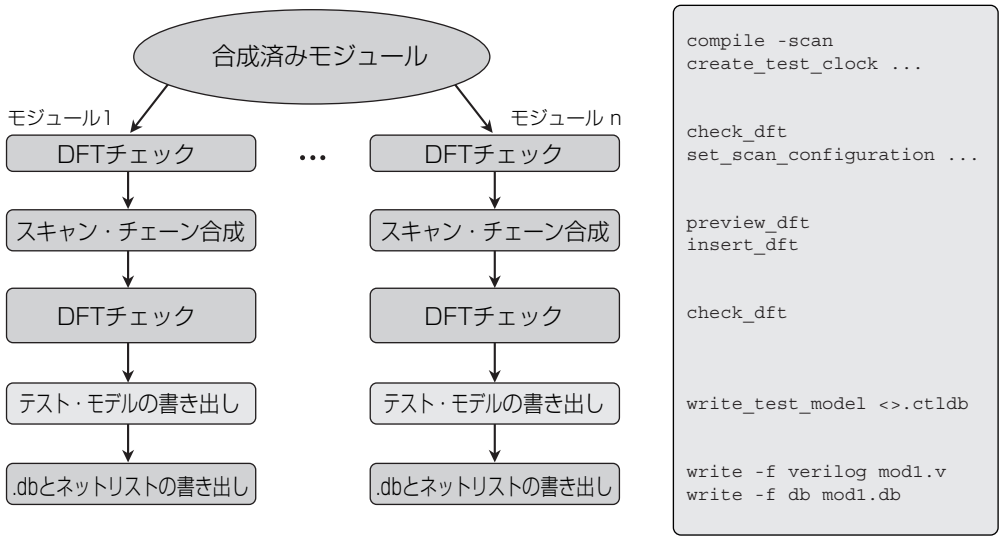

図4: モジュールレベル・スキャン合成フロー

トップレベルでのチップ・アセンブリの際には (図5)、すべてのサブ・モジュールのテスト・モデルと、トップレベル設計記述がDFT Compilerに読み込まれます。これにより、各サブ・モジュールのフル・ゲートレベル・データベースの読み込みが回避されます。トップレベル設計のスキャン設計ルール・チェックは容易に実行することができます。また、トップレベル・スキャン・チェーンのバランスと構築も容易に実現できます。この手法を用いることにより、既存のユーザはDFT Compilerを使った同じボトムアップ・フローを引き続き採用することができるだけでなく、処理時間の性能とメモリ使用量を飛躍的に向上させる新しい機能を活用できます。

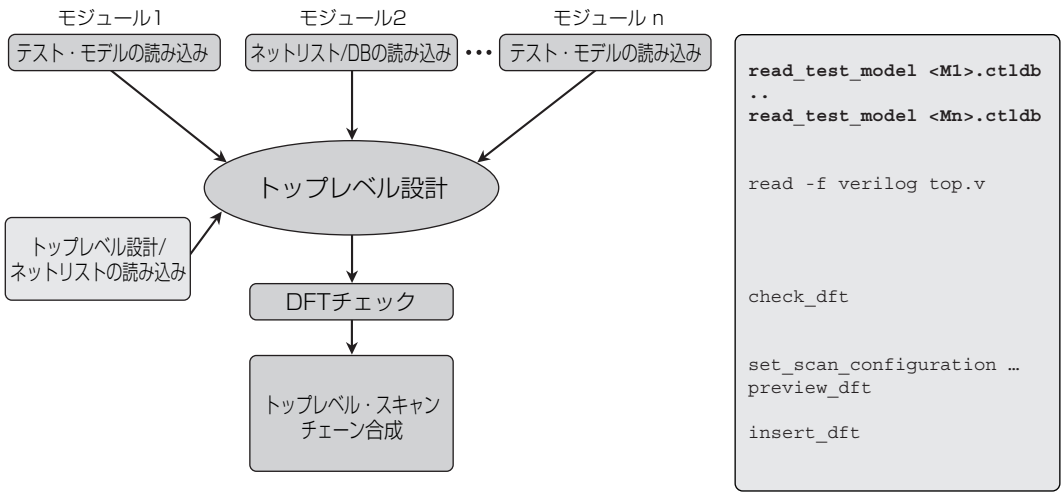

図5: テスト・モデルを使用したトップレベル・スキャン合成フロー

インターフェイス・ロジック・モデル(ILM)とテスト・モデル

インターフェイス・ロジック・モデル(ILM)は、PrimeTime、Design Compiler、およびPhysical Compilerの階層設計フローに対する正確で豊富なモデル生成ソリューションを提供しています。ILMは、モデル生成の体系的手法に基づくものです。このアプローチでは、ブロックの元のゲートレベル・ネットリストが、ブロックのインターフェイス・ロジックを含む別のゲートレベル・ネットリストによってモデル化されます。インターフェイス・ロジックには、I/Oポートからインターフェイス・レジスタと呼ばれるエッジトリガ・レジスタまでのすべての回路が含まれています。インターフェイス・レジスタに至るクロック・ツリーは、ILMに保存されます。ブロックのレジスタ間のパスにのみ含まれるロジックは、ILMに保存されません(図6)。

ILMの利点は以下のとおりです。

- 修正のないインターフェイス・ロジックを保存することにより、ILMは元のデザインを忠実に表現したものとなります。ILMは抽象化せず、バウンダリ・タイミングのモデル化に不必要なものを排除します。レイアウト後のフローの一般的な設計とテクノロジの場合、ILMは元のブロック・タイミングの±10ps以内で保持します。
- インターフェイス・ロジックが維持されるため、元のネットリストと比べてILMのタイミング特性の違いは容易にデバッグ可能なことです。クロックとデータ・パスは修正されずに保存され、セルとネット名は同一であるため、不一致の原因を容易に排除できます。
- 設計のインターフェイス・ロジックの識別が回路トポロジの解析によって実行される体系的な処理であるため、ILMを使用したモデル生成は高速です。
- ある一定の種類の回路では、ILMによりモデル・サイズが大きく減少しないケースもあります。たとえば、純粋な組み合わせ論理ブロックや複数レベルのタイム・ボローイングを持つラッチベースの回路などです。このような回路のインターフェイス・ロジックには元の設計が多く含まれる傾向があります。

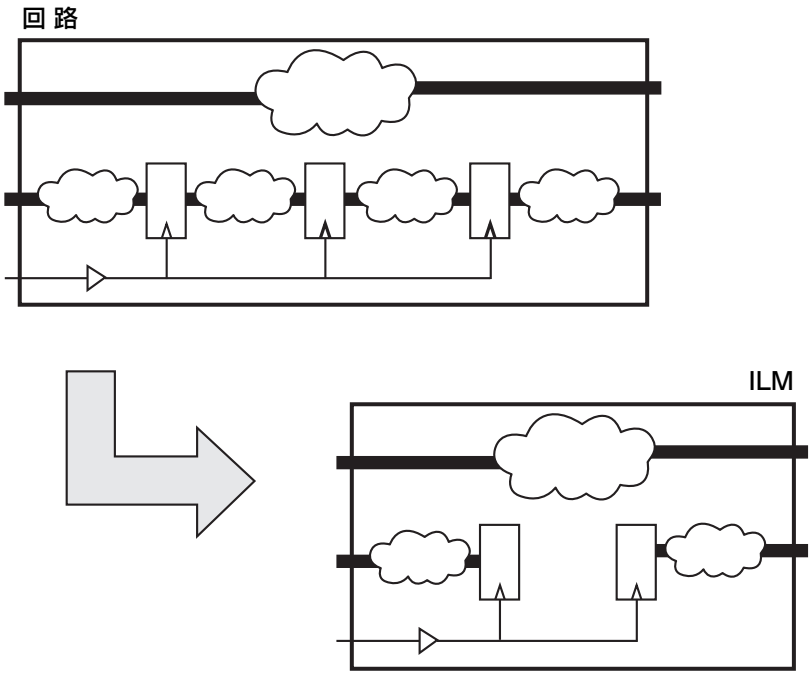

図6: インターフェイス・ロジック・モデル(ILM)

ILMとテスト・モデルは、大規模設計の既存の階層設計フローを維持し、タイミング、物理制約条件、およびテストのすべての条件を満たすために、Design CompilerとPhysical Compilerの環境に統合されています。ILMはデザインから抽出され、ブロックのテスト情報と物理的な位置とともに保存されます。図7に、Physical Compilerでのテスト・モジュールとILMの統合、およびテストと物理情報をもとにILMを生成するプロセスを示します。サブモジュールは、Physical Compilerで一般的な物理合成とワンパス・スキャン・オーダリングの工程を通ります。次に、サブモジュールILMを抽出して、サブモジュールのポートのバウンダリ・タイミング情報と物理情報、およびスキャン・チェーンとその他のテスト関連情報を含むSynopsys dbフォーマットとして保存できます。

トップレベルでは、トップレベルのフロアプランとともにすべてのILM (db) をPhysical Compilerに読み込めます。トップレベル・デザインはハード・マクロ、グルー・ロジック、およびILMの組合せからなっています。同じ物理合成設計フローをトップレベル設計に適用し、物理位置に基づいたスキャン・チェーン・オーダリングを行うことが可能です。この物理設計の階層的なアプローチとタイミングの最適化により、迅速なタイミング収束を達成するとともに、設計期間の短縮と設計のやり直しの削減を実現し、生産性を飛躍的に向上させます。

同じフローをDesign Compilerで採用することにより、タイミングおよびテスト情報だけを有するサブモジュールILMを抽出し、これらのサブモジュールILMをトップ・レベルで統合して、すべてのタイミングおよびテスト制約条件を満たす階層スキャン・チェーン合成を実行することができます。

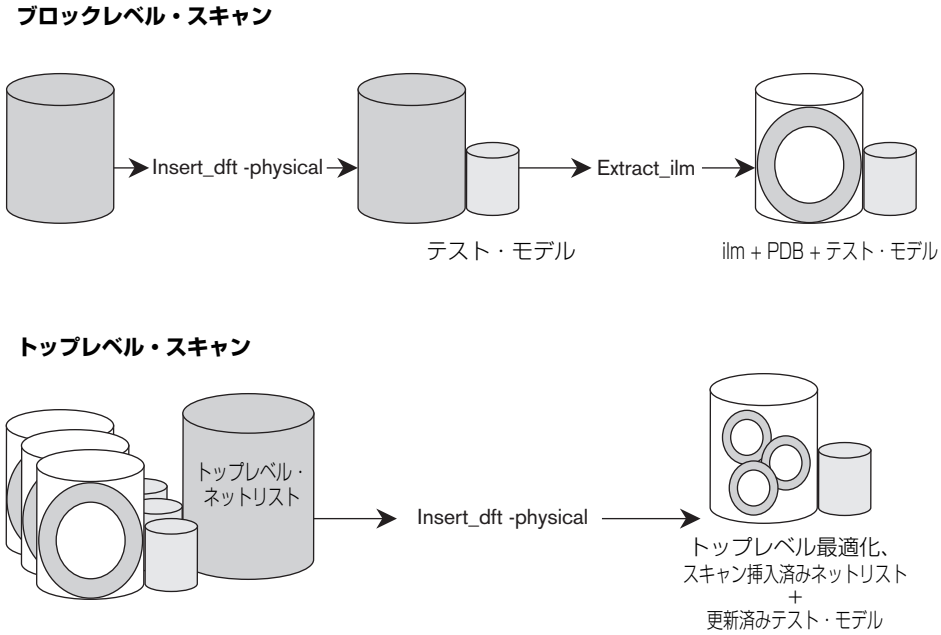

図7: Physical Compilerを用いたILMおよびテスト・モデルの抽出フロー

結論

ASICおよびSoC設計の超大規模化の進展に伴い、設計者とシステム設計者は階層ボトムアップ・スキャン合成フローを採用し、タイミングとDFTの収束を迅速に実現する必要があります。しかし、サブモジュールの集積度も増大していることから、大規模なブロックのスキャン合成を実行するには容量と処理時間が最大の課題となっています。大規模設計に対する現在の課題は、スキャン設計ルール・チェックとスキャン・チェーン設計をチップ・レベルで実行して、設計のやり直しを最小限に抑えてタイミング収束を達成することです。

DFT Compilerのテスト・モデルを使用した階層スキャン合成の導入により、設計者は現在のソリューションに比べて大幅な性能改善を実現できます。一般的な数百万ゲートのチップレベル設計では、テスト・モデルを使用した階層スキャン合成アプローチにより、処理時間が7倍以上、メモリ使用量も3倍の向上を実現しました (図8)。

DFT Compilerでテスト情報が自動的にテスト・モデルに抽象化されるため、設計者はフローを意識する必要がありません。強力なモデリング技術により、階層スキャン合成の性能の飛躍的な向上と完全な自動化が実現されるため、容易な導入が可能となり、設計生産性も向上します。設計者とシステム設計者は、高度な階層DFTフローを採用して複雑なDFTを克服することができます。

テスト・モデルとILMの統合により、Design Compilerのすべてのテストおよびタイミング制約条件を満たす階層チップレベル・スキャン合成の完全なソリューションが実現されます。Physical Compilerの環境では、ILMとテスト・モデル・テクノロジを使用して、超大規模設計に対応した階層ワンパス・フィジカル・スキャン合成を実行できます。これにより、設計のやり直しを最小限に抑えて、テスト、タイミング、および物理制約条件を満たし、生産性を飛躍的に向上させることができます。

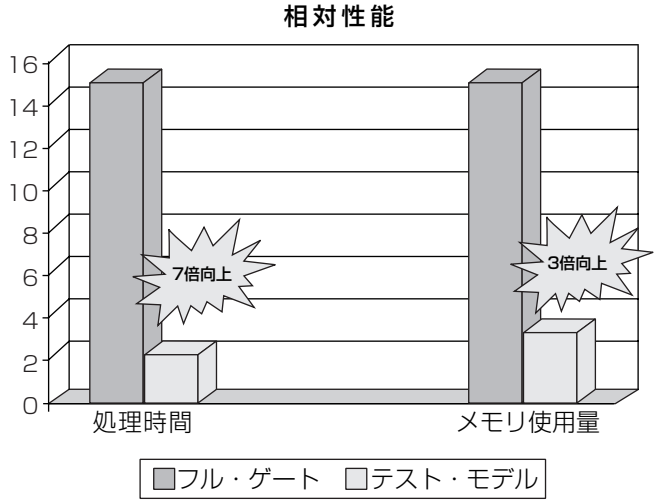

図8: 階層スキャン合成でテスト・モデルを使用した結果

お問い合せ先：

日本シノプシス株式会社

〒163-0420 **東京都新宿区西新宿**2-1-1 **新宿三井ビルディング20F** TEL.03-3346-7030**(代)** FAX.03-3346-7050

〒531-0072 **大阪府大阪市北区豊崎**3-19-3 **ピアスタワー13F** TEL.06-6359-8139**(代)** FAX.06-6359-8149