特集

システムLSI設計の検証技術

Verification Technology for System LSI Design

あらまし

システムLSIの論理規模増大への対応のため，設計品質の観点からLSI設計の検証技術について述べる。2000年には，ランダム論理は300万～500万トランジスタの設計規模となり，設計不具合も現在の1.5～3倍増加すると予測される。LSI設計における設計不具合の約70%はテスト項目の組合せ不足であり，設計品質の面から新しいLSI設計検証技術が必要となる。大規模化に対応して形式的検証を導入することは，論理シミュレーションでは検出困難な設計不具合を発見でき効果が期待できるが，本格導入のためには処理規模の拡大と検証用モデルの自動抽出が課題である。原因-結果法によるテスト項目生成は，仕様レビューと抽出した項目組合せの網羅性で効果があるが，LSI設計に適した仕様項目記述方法と項目間制約方法の改良が課題である。

Abstract

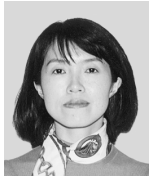
Further increases in the scale of System LSI require a new design verification technology that improves LSI design quality.

By the year 2000, random logic designs will contain from 3 to 5 million transistors and 1.5 to 3 times the number of bugs found in current designs. About 70% of all design bugs come from insufficient LSI design verification.

New verification technologies such as Formal Verifications based on model checking and test case generation based on Cause-Effect Graphing have been investigated in actual system LSI design verifications. However, in Formal Verification, design size limitation and RTL model generation with reduction and abstraction are serious problems. In Cause-Effect Graphing, the problems are how to specify criteria and extract LSI specifications.

木村雅春（きむら　まさはる）

1969年岡山県立津山工業高校卒。同年富士通入社。以来マイクロプロセッサ/マイクロコントローラの開発を経て，LSI設計評価技術の開発に従事。
システムLSIソフトウェア部評価システム開発部



久門由紀（くもん　ゆき）

1984年日本女子大学家政理学部理Ⅱ学科（生物）卒。同年富士通入社。以来LSIのCAD開発を経て，LSI設計の評価システムの開発に従事。
システムLSIソフトウェア部評価システム開発部

安倍健志（あべ　けんじ）

1987年東京理科大学工学部経営工学科卒。同年富士通入社。以来キーボード設計支援システムの開発を経て，LSI設計評価技術の開発に従事。
システムLSIソフトウェア部評価システム開発部

ま え が き

システムLSIが高機能化するにつれ設計規模が増大しており，その設計手法も設計規模の増大に対応して進歩してきた。大規模化の対策は主として設計期間短縮に重点が置かれているが，設計品質の面からも大規模化への対応に備える必要がある。

そこで本稿では，設計規模増大への対応として設計品質の観点から，LSI設計規模と設計品質の予測を示す。これに対処するため，論理シミュレーションによらない形式的検証および機能テスト項目の抽出とテストケース生成を行う，LSI設計の検証技術について紹介する。[(1)-(6)]

設計規模の増大と課題

● 設計規模

LSIの集積素子数はプロセッサ系LSIで2000年に1,000万～1,500万トランジスタ/cm²になると予想される。プロセッサ系のLSIの富士通製品における設計規模の推移を**図-1**に示す。[(注1)] チップ周辺面積を考慮したチップ実設計規模としては700万トランジスタ，そのうちランダム論理回路の設計規模は300万～500万トランジスタになると予想される。論理設計と設計検証の工数はランダム論理回路設計に費やされ，いろいろな問題の原因となる。設計不具合は，経験的には素子数比の0.2 ～ 0.3乗に比例して増加し，700万トランジスタのLSI設計では，現在よりも1.5～3倍の設計不具合の増加が予想され，検証能力の拡大が課題となる。

● 検証内容

検証内容の課題検討のため，あるLSI開発における設計不具合の内容分析を行った(**図-2**)。この図から項目組合せミスが設計不具合の約半数であり，単純ミスを含めて論理修正の約70%はテスト項目の組合せに関連していると言える。

仕様・項目の組合せミスは，仕様ミス，条件組合せ不足，設計者間の連絡忘れなどであり，単純ミスは仕様勘違い，確認漏れなどであり，これらのことから検証内容として網羅性が課題となる。そこで，設計規模の増大に対する以下の対策について次章以降で述べる。

(1) 検証能力拡大のための新しい機能検証手法

(2) 完全な機能検証を目指した検証の網羅性のための形式的検証およびテスト項目組合せ生成

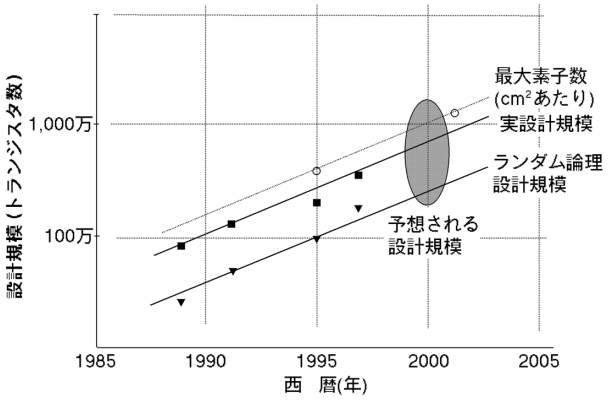


図-1 設計規模の推移
Fig.1-Transition of random logic design size.

(注1) 素子数はチップサイズによる。グラフはパッドを含む現実的チップ設計の素子数の推移を示す。

検証能力拡大

LSI開発における検証技術マップを**図-3**に示す。

● システム検証

実際にシステムを動作させ，エミュレータ，ボードなどを使用してシステムとしての確認が行われる。これはソフトウェアを含む実時間動作であり，大規模かつ非常に高速な検証方法であるが，設計終盤で論理がほぼ固定してから行われるという制約がある。

● 高速論理シミュレーション

これは設計期間の短縮であるが，結果的に機能検証の機会が増えるので，設計品質の向上も期待される。

(1) シミュレーションの大規模化・高速化

サイクルベース・シミュレータ，ハードウェア・アクセラレータ，Cモデル記述併用などにより高速化される。

(2) HDL(ハイレベル設計言語)記述の採用

ビヘイビアまたはRTL(レジスタトランスファ言語)を用いたシミュレーションにより高速化を図る。

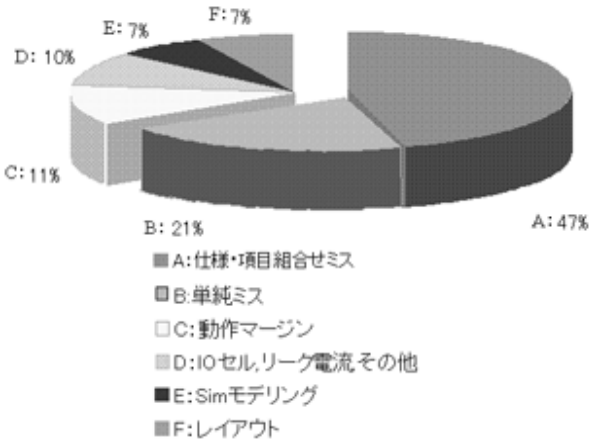


図-2 設計不具合の内容分析
Fig.2-Factor of logic design bugs.

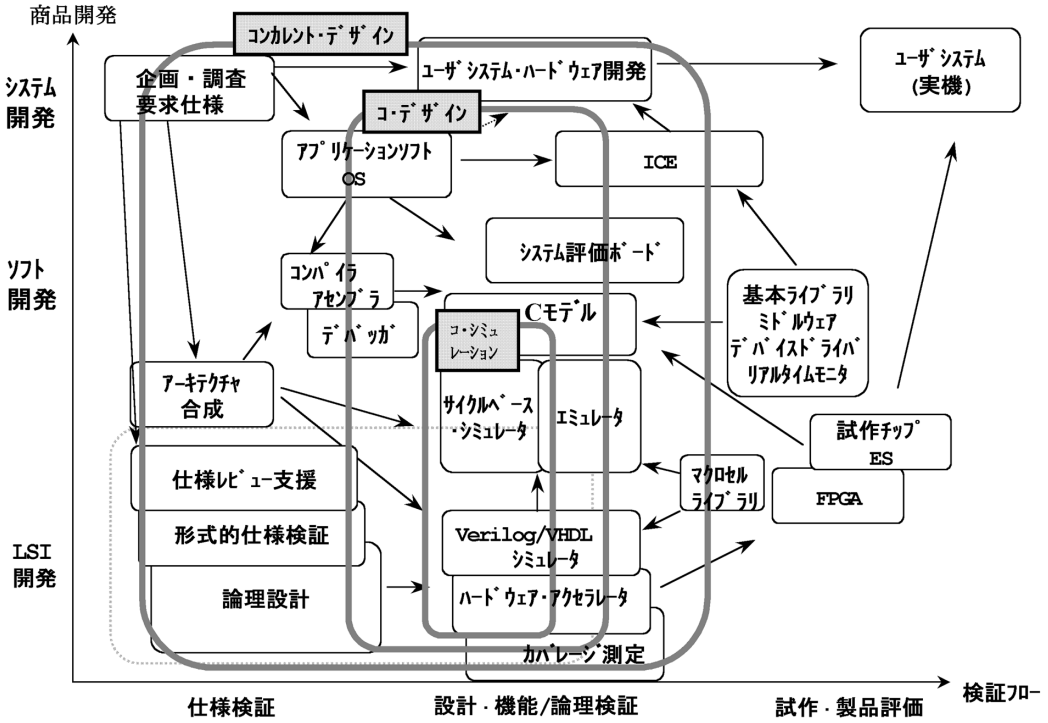


図-3　検証技術マップ
Fig.3- System verification flow.

●　シミュレーションモデル記述カバレージ測定

検証ではないが設計品質を直接に評価するものとして，HDL(RTL)記述のカバレージ測定がある。一般にテストされていないところにバグがあるという経験則から，HDL(RTL)記述を論理シミュレーションによりテストパターンを入力しカバレージ測定をすれば，客観的に設計検証の程度が分かり，設計不具合の削減に効果が大きいことが予測される。現状のパスカバレージ，状態遷移のパスカバレージなどまだ機能不足であるが，現在すぐに活用できる技術である。

●　基準仕様作成

図-3におけるコンカレント・デザインあるいはコ・デザインは，単にソフトウェアとハードウェアの協調設計・並行作業ということだけでなく，品質は設計の上流で作り込むという観点から，基準仕様としてシステム動作のモデル記述や基準テストデータを作成することが望まれる。ここでLSI機能検証に必要となる「仕様」とは必ずしもLSI仕様書を意味していなくて，全体動作として期待する動作・動作条件・前提条件・例外条件などの仕様のモデル記述である。例えば，

(1)　動作要求信号が出ると必ずいつか許可される，
(2)　同時に動作要求は出ない，
(3)　動作要求毎に毎回許可される，
(4)　デッドロックがない，

などの暗黙の前提事項である。

形式的検証

従来の論理シミュレーションによる機能検証は，長いテストパターンを作り延々とシミュレーションを行わなければならず，設計規模が大きくなると時間・検証内容ともに十分というまでが大変だった。そのため，論理シミュレーションに代わる検証方法としてモデル検査(Model Checking)を行う形式的検証が注目されている。

その特徴は，チェックした特性(仕様)では他にバグがないことが完全に保証され処理も速いことである。これで見つかるバグは非常に複雑な条件の組合せであり，論理シミュレーションでは発見困難なことが多い。

現在，富士通が開発した形式的検証ツール“ BINGO ”を使用して，形式的検証のLSI設計検証への適用評価を行っている。

●　モデル検査

形式的検証という言葉はあまり馴染みがないが，LSI設計向きに分かり易くいえば「検証対象モデルの外面から，数学的・論理的に所定の動作を証明すること」である。**図-4**を用いて形式的検証の説明を行う。

(1)　モデルの状態遷移記述

まず検証すべきLSIのRTL記述から状態遷移記述モデルを抽出する。

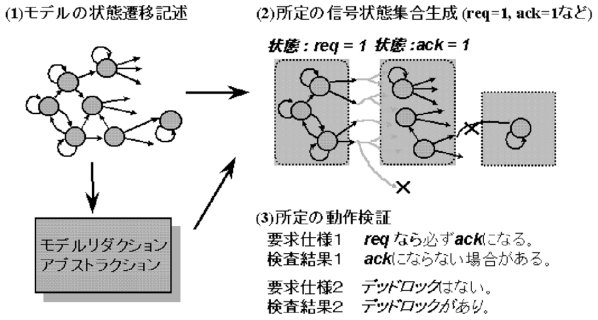


図-4　形式的検証
Fig.4-Formal verification flow.

(2)　検査すべき仕様決定

検査すべき所定の動作(仕様)と検査表現を決定する。

(3)　状態集合生成と所定の動作の検査

状態遷移記述モデルから所定の仕様動作(プロパティとも言う)を満足する状態集合を求め，集合間の論理演算(時相論理演算と言う)によりパス探索を行う。

もしも所定の動作を満足できない遷移パスが見つかれば，その状態遷移を引き起こすテストパターンを生成して，論理シミュレーションで確認することになる。

●　基準テストパターン生成

形式的検証は，モデル検査以外にも基準となるテストパターンの生成ができるという特徴がある。つまり検査済の正しい論理モデルを使用して，所定の動作をするための状態遷移を起こすテストパターンを生成すれば，これが基準テストパターンとなる。これを論理シミュレーションに使用すれば，仕様どおりに論理が実現されたかどうかの論理設計の検証に使用でき，従来の人手による長大な機能確認用テストパターン作成を不要にする効果がある。

●　形式的検証の問題と課題

形式的検証の問題と課題は，処理規模の拡大，モデルリダクション(モデル縮退)，アブストラクション(モデル抽象化)および網羅性である。

(1)　処理規模の拡大

状態フリップフロップ数にして数百個前後が現在の処理規模であり，状態数としては非常に大規模(2^{500} = 約10^{150})であるが，これでも処理規模をオーバすることがよくあり，現実の設計規模にまだ及ばない。そのため，モデルリダクションおよびアブストラクションにより処理規模を拡大する研究が行われている。

(2)　リダクション/アブストラクションモデル作成

論理モデルの検証規模を小さくするため，モデル作成におけるモデルリダクション，アブストラクションが必要である。この際に，検証すべき論理モデルの動作および仕様の理解が必要になり，これがネックになり形式的検証での労力と時間を費やしている。そのためモデル作成でのリダクション，アブストラクションの自動化が大きな課題である。

形式的検証の適用で残っている課題として，モデル検査した検証項目の網羅性がある。検査した特性は完全に保証されるが，検査すべき特性の組合せ漏れ，すなわち網羅性についてはまだ良い方法はない。

●　適用評価例

実際のLSIで適用評価した例としては，メディア処理向けプロセッサLSI(素子数：200万トランジスタ)がある。このLSIは，5本の内部バスを有しバスマスタが5種類存在し，高速転送のためバス間の転送はパイプライン動作するという非常に複雑なバス制御を行っている。元のRTLからバス関係動作の状態遷移記述を抽出したが，フリップフロップが約5,000個と規模が大きかった。このためモデルリダクションを行い，フリップフロップを数百個まで減少させた。この検証用モデルを使用して内部バス動作について形式的検証を適用した結果，すでに見つかっているバグが形式的検証で再現できるということが分かったこと，別の新たな1件のバグを発見し，その論理修正も正しいということを検証できた。

形式的検証は，結果が保証されるという画期的な検証手法であるが馴染みのない手法であるため，論理設計の参考となるよう，形式的検証用モデル作成のポイントを以下に示す。

【モデル作成でのVerilog文法の制約】

(1)　論理合成可能なRTL記述
(2)　信号値: 0，1のみ
(3)　ゲートレベル記述適用不可
(4)　エッジ型フリップフロップ使用
(5)　単一クロック使用
(6)　ゲーテッドクロック(代入文の右辺の要素にクロックを使用)不可
(7)　双方向バス記述，その他若干の文法上の制約

【リダクション，アブストラクションモデルの作成】

(1)　データパスはレジスタ多数のため形式的検証に不適当
(2)　機能的に等価な上位機能表現のモデル記述へ書換え
(3)　レジスタ宣言されたフリップフロップ数が処理規模の目安

上記の説明は論理シミュレーションとして問題になるものがあるが，これらの制約は使用する形式的検証ツールに依存しており，今後改善される。また処理規模の制

約から，LSIの検証手法としては形式的検証は，当分論理シミュレーションと併用されると予想される。

テスト項目の組合せ生成

LSI設計における機能検証では，そのテスト項目の組合せ(テストケース)が設計品質に大きな影響を与える。

● 原因-結果法によるテスト項目抽出

原因-結果法(Cause-Effect Graph)はブラックボックス・テストの一種であり，条件項目(原因)と所定動作(結果)とこれら要因項目間の論理関係を記述した原因-結果グラフを作成し，所定動作に必要な条件の項目の組合せをテストケース1，2，3・・として作成する方法である。この要因項目の抽出・登録，項目組合せの数を減らすための条件要因間の制約・相関の論理的関係の入力が，仕様レビューになり，この段階で仕様の曖昧さ，仕様の不備などが見つかることがあるという特徴がある。

抽出したテスト項目(1〜14，23)とその原因-結果グラフを**図-5**に示す。要因項目間の制約関係は代表的なものとして，排他的(E)，ある要因が成立するときは他の要因も同時に成立する(R)，AND・ORなどの論理関係を使用して要因項目間の制約の入力を行うが，これは論理図作成と似た作業である。

● 原因-結果法適用の課題

原因-結果法は，もともとハードウェアをソフトウェアでテストするために考案され，ソフトウェアテストから経営問題まで適用できる手法であるが，LSI設計に適用するには組合せの考え方が，ソフトウェアテストとは若干異なる。

ソフトウェアの場合は，実験計画法に準じて組合せはどこかですべての組合せが行われていればよいが，LSIの場合は，複数の内部の状態遷移マシンにより動作が同時に実行されるので組合せはもっと増える。例えば，図-5の原因-結果グラフにおいて，14要因項目に対して6個の制約をつけた結果，「どこかですべての組合せをする」方式では最少の7通りであったが，ハードウェアのテストとして生成されるテストケースは54通りが期待される。

原因-結果法は組合せ生成が系統的に行えるという特徴があるが，LSIの設計検証に適用するため，組合せ削減のため制約の指定方法の検討，状態遷移の入力，項目入力方法の改善などの課題がある。

今までは項目入力手段がネックとなって使用しづらかったため，**図-6**に示すような原因-結果法によるテストケース生成のためのテスト項目入力支援ツールを開発した。現在，その出力機能の開発と適用評価を行っている。

む　す　び

LSI開発における設計規模の増大に対する設計品質の観点から，形式的検証，テスト項目生成について紹介した。いままでの評価結果からはかなり有効な検証手法であると言えるが，本格的にLSI設計に導入するにはまだ課

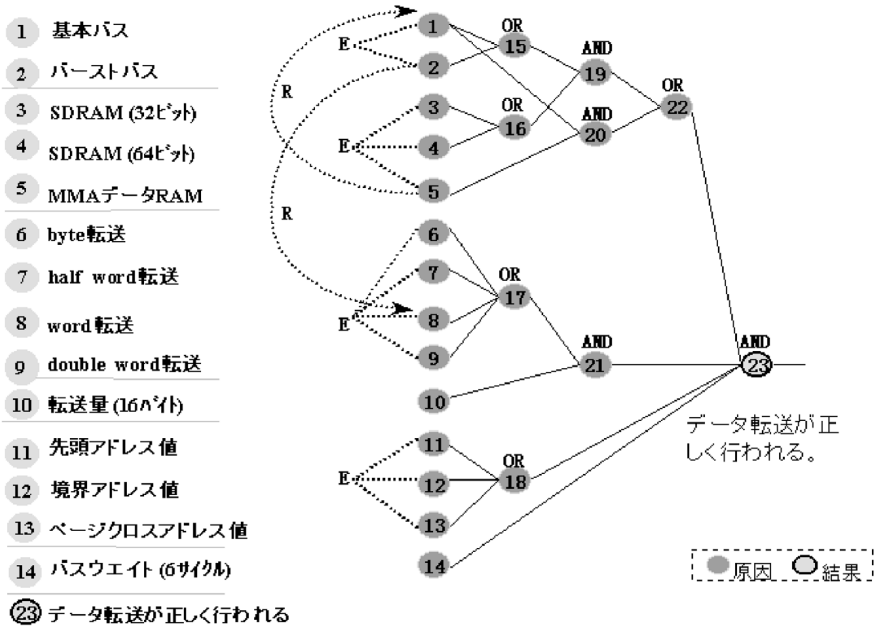


図-5　テスト項目と，その原因-結果グラフ
Fig.5-Test items and Cause-Effect Graph.

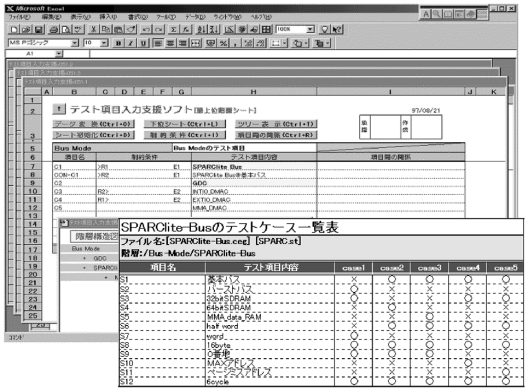


図-6　テスト項目入力支援ツール
Fig.6-Test item entry tool.

題が多い。論理設計者は長い間，「完全な設計」ができる検証ツールを求めており，その解答の一部がここで紹介した検証技術である。

設計品質の向上により開発期間の短縮を図り，顧客に満足して頂けるように努めたい。

参考文献

(1)　長尾ほか：ソフトウェアテストの技法. 1980，初版，近代科学社.

(2)　石井：ソフトウェアの検査と品質保証. 1986，初版，日科技連出版社.

(3)　藤田ほか：特集「BDD(二部決定グラフ)－幅広い応用範囲をもつ論理関数処理技術－」の編集にあたって. 情報処理，**34**，5，pp.593-630(1993).

(4)　平石ほか：特集「論理設計の形式的検証」の編集にあたって. 情報処理，**35**，8，pp.710-750(1994).

(5)　－：特集「システムASIC」. 日経エレクトロニクス，NO.697，pp.67-85(1997年9月1日).

(6)　岩下ほか： Forward Model Checking Techniques Oriented to Buggy Designs. ICCAD，1997.