

1 チップCPUプロセッサの設計検証(1)

5N-7

CPUプロセッサの機能モデルの開発

服部孝, 村田裕, 宮内信仁, 清水徹, 星直之, 上野仁
三菱電機(株)

1 はじめに

ビジネスコンピュータ用1チップCPUプロセッサを開発した。このチップは、0.8 μ m CMOSプロセスを用いたフルカスタム設計手法で設計され、約170万Tr.規模のものである。

チップ開発は、内蔵される機能の規模・複雑さが増すにつれ、チップ作り直しのリスクが高まる方向にある。このため、開発工程の大幅な後戻りを避けるために、より高品質の設計検証がチップ設計初期から要求されている。

今回の開発では、トップダウン設計の観点から、チップの機能仕様および外部仕様を早期に検証するためにハードウェア記述言語であるVerilog-HDLを用いてCPUチップの機能モデルを開発した。

本稿では、CPUチップ開発で行われた設計検証のうち、機能検証を中心に機能モデルの概要と機能モデルを利用した検証の効果について報告する。

2 機能モデルを用いた検証の狙い

設計検証には、1)設計仕様の正当性、2)仕様に基づきトランジスタ/ゲートレベルで設計される論理図の正当性、の2つを検証する目的がある。しかしながら、論理図をベースとした論理モデルを仕様の検証に利用するのではチップ検証の開始が遅れてしまい、また、論理検証自体もシミュレーション速度や図面改訂、不具合部分の特定に時間がかかり、検証効率が悪いのが現状である。

そこでCPUチップ単体機能に関わる設計検証として、機能/論理/一致/システムの4つの検証レベルを設定し、設計仕様の早期の

検証と論理検証の効率化を図るために、シミュレーションが速く設計・改訂のターンアラウンドが短い機能モデルを、各レベルの検証に適用することを考えた。

図1にチップ設計時に行われた検証の概略イメージを図示し、以下にそれぞれに期待された役割とそこでの機能モデルの利用方法を簡単にまとめる。

1) 機能検証

- 機能モデルを利用し、仕様、性能不具合の早期検出。
- マイクロコードのデバッグ。
- 完成度の高いテストプログラムの作成。

2) 論理検証

- 機能仕様とマイクロコードは機能モデルで確認されており、論理検証では論理回路そのもののデバッグに注力。

3) 一致検証

- 同一テストプログラムによる機能/論理シミュレーション結果をマシンサイクル毎に比較することで検証を行う。

4) システム検証

- 周辺チップを含むCPUボードモデルに機能モデルをパーツとして組み込み、システムとしての観点からCPUチップのテストを行う。

3 機能モデルの概要

2で挙げたように機能モデルは、仕様の検証だけではなく、一致検証に耐え得るような動作を実現し、また、チップ単体よりも大規模となるシステム検証に対しては、高速なシミュレーション性能を保証しなければならない。

背反する記述の詳細さとシミュレーション性能を考慮し、次のようなCPUチップ機能モデルの開発を行った。

3.1 記述方針

- チップ内ブロックを単位として記述し、ブロック間インターフェース、内部レジスタ・メモリ・バス等の主要ハードウェアリソースを論理回路に合わせる。また、チップインターフェースも合わせる。
- 動作は、内部クロックに同期したレジスタ転送レベルの記述とする。

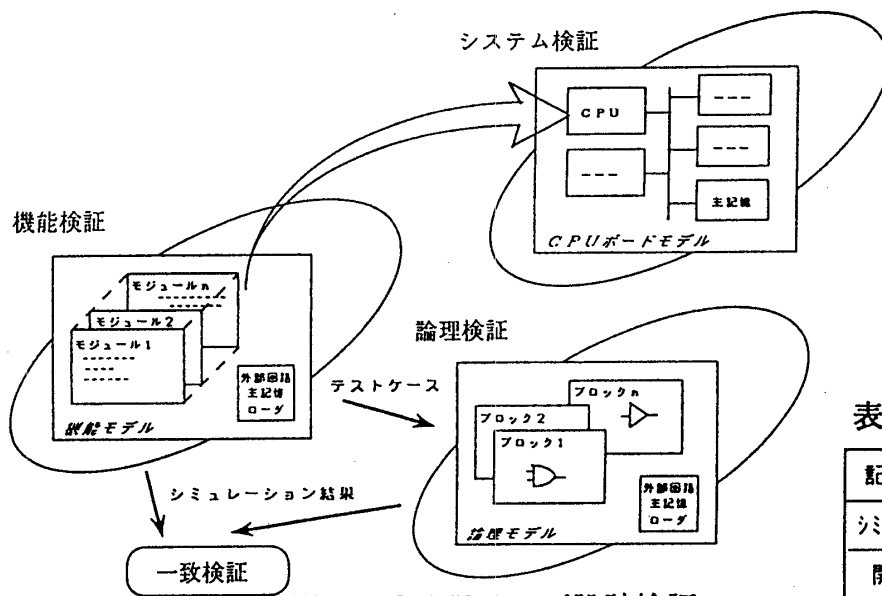


図1 CPUチップ設計検証

表1 機能モデル

記述ステップ数	37 K step
シミュレーション速度	論理モデルの10倍
開発人工	65 人月

3.2 機能モデルの構成

記述方針に従って、パイプラインのステージ毎に分割された6つのブロック単位にモジュールを構成した。またCPU本体とは別に、主記憶、割込みコントローラ等を含む周辺回路の動作を模擬するモジュールや、テストプログラムの実行制御を行うモジュールも用意した。

3.3 開発の実際

機能モデルの開発は、第三者検証的な効果を期待して、仕様書をベースに、ブロック毎に論理設計者とは別の担当者によって行った。機能モデル自体のデバッグは、従来機で実績のある診断プログラムの流用で効率化を図った。ソースコードは、UNIXのSCCSで管理した。

1次版の機能モデルが開発された時点では、記述者や記述対象となる機能によって記述構造が大きく異なり、シミュレーション性能的にもばらつきがあるものであった。そこでまず、性能評価ツールを用いて機能モデルの各ステップの出現頻度を測定し、この結果を元に冗長なステップの削減やイベントの統合化を行った。次に、基本的な機能に関して表現の異なる記述のものを抽出し、それぞれのシミュレーション速度を比較することで記述の最適化を行い、シミュレーション性能の向上を図った。

シミュレーション性能向上のためのチューニングや一致検証からのフィードバックを適宜行い、最終的に表1に示す機能モデルとなった。

4 機能モデルを用いた検証の効果

機能モデルを用いた機能検証として、次のような成果が上げられた。

- ・仕様に関する不具合は、機能/論理検証で報告されたもののうちの9割以上を機能検証段階で検出でき、大きな後戻りの発生を抑える効果があった。
- ・マイクロコード、テストケースに関する不具合は、機能検証において共にほぼ100%を検出でき、論理検証では、論理回路の不具合追求に注力することが可能となった。論理デバッグ期間の短縮に効果があった。

また、次のような問題から不具合の検出漏れが発生し、検証効率を落とす場合があった。

- ・don't careの取扱い等、機能/論理でのモデリングの違いによる不具合の見逃し。

5 まとめ

170万Tr.レベルの論理規模を持つプロセス開発において、設計検証の効率化のため機能モデルを利用した検証を実施した。今回の機能モデルは、チップピンおよび主要内部ハードウェアリソースのみを、マシンサイクル毎に論理モデルの動作と一致させ、それ以外の部分は記述を簡略化することでシミュレーションの高速化を図ったものである。

結果として仕様の検証だけでなく、一致検証やシステム検証を早期に立上げることができ、全体として論理検証の効率化を図ることができた。