審 査 の 結 果 の 要 旨

論文提出者氏名　西原　佑

本論文は、Formal Verification of High-Level Design Based on Control/Data Separation（高位設計に対する制御と演算の分離による形式的検証手法）と題し、SoC（System on a Chip）や組込み機器設計に対する高位設計において、設計の正しさを数学的に証明する形式的検証技術に関し、設計対象の動作を制御部分と演算部分（データパス部分）に分離して効率的に解析する新規手法を利用することで、実用規模の高位設計の形式的検証を可能とする技術が示されており、英文で７章から構成されている。

第１章は、Introduction（序章）であり、研究の背景と目的を述べている。高位設計の現状を設計の流れと設計記述言語の観点から整理し、高位設計における形式的検証の重要性が示されている。さらに、効率的な検証の上で重要である制御とデータ演算を分離して解析する技術の基本を示し、論文全体で示される検証手法の考え方を整理している。

第２章は、Preliminaries and Basic Notions（背景と基本技術）であり、高位設計における形式的検証技術の基本と、高位設計記述の計算機での表現やその解析技術に関し、現状技術の整理を行っている。本論文で示される形式的検証技術は、制御部分とデータ演算部分の分離を行いながら、モデルチェッキングと呼ばれる網羅的な設計解析を効率的に行う手法、ならびに設計記述を網羅的にシミュレーションし等価性も検証できる記号シミュレーションなどを基盤とし、それらの拡張や階層的な利用により実現されている。

第３章は、Multi-Level Bounded Model Checking（多段限定モデルチェッキング）であり、従来から提案されている限定モデルチェッキングを説明した後、その問題点を整理し、それを克服する多段限定モデルチェッキング手法を提案・評価し、その有効性を示している。通常のモデルチェッキングは初期状態からエラー（バグ）のある状態への状態遷移列全体を検索しているが、限定モデルチェッキングではその検索の際の状態遷移列の長さを制限することで検証を効率化し、より大規模な設計検証を可能としている。しかし、限定モデルチェッキングでは、解析の際の状態遷移列の長さを超える反例は原理的に見つからないため、複雑なバグは発見しにくい。そこで、状態遷移列の探索を１回だけでなく、複数回に分けて行うことで、より長い反例も見つかるようにした多段限定モデルチェッキング手法を提案・評価し、有効性を示している。単純に多段にするだけでは複数の探索した状態遷移列が接続せず、無駄な探索も多くなるため、探索された状態遷移列の最初と最後の状態を管理し、それらが連結するように探索法を変化させる手法などが導入されている。

第４章は、Equivalence Checking with Synthesizing Designs onto Identical Datapath（同一データパスへの合成を利用した等価性検証）であり、与えられた２つの設計記述を同一のデータパス上での実行シーケンスに変換することにより、２つの設計記述の等価性判定をそれらの制御部分のみの解析で実現できる手法を提案・評価し、その有効性が示されている。形式的検証手法では、解析すべき状態数が膨大となり検証できなくなる「状態

爆発」と呼ばれる問題により、扱える設計規模が制限されている。特に設計のデータパス部分（演算を実行する部分）は、大きなレジスタなどの記憶素子や複雑な演算器があり、状態数が制御部に比較し圧倒的に大きい。そこで、与えられた２つの設計記述を同一のデータパス上での実行にマッピングする（コンパイル）することで、動作の等価性をデータパスへの制御信号のシーケンスの等価性に帰着させることができる。この考え方に立ち、既存のコンパイラ技術を利用して実際にコンパイルし、その結果から自動的に検証すべき制御シーケンスを生成する手法を新規の考案しており、例題での評価結果から、通常の等価性検証と比較し、大幅性能向上が示されている。

第５章は、Formal Verification of Hardware/Software Co-Design with Translation into FSMD（FSMD への変換によるハードウェア・ソフトウェア協調設計の形式的検証）であり、ハードウェア・ソフトウェア協調設計に対しその通信部分も含めて検証を形式的に行う手法が示され、評価によりその有効性が実証されている。従来はハードウェア、ソフトウェアを別々に検証するものがほとんどであり、両者の通信部分も適切にモデル化しなければならないことから協調設計の形式的検証の提案はほとんどなかった。そこで、実際に広く利用されている通信部分の実装を基にそれをモデル化し、協調設計全体に対する検証モデルを自動的に生成する手法を新規に考案している。具体的には通信における同期を自動的に認識し、それをもとに協調設計記述の融合・簡単化を実現している。

第６章は、ExSDG: Design Representation for Efficient High-Level Design Verification（高位設計の効率的な検証のためのデータ構造：ExSDG）であり、前章までで示された検証手法を実装し、実際の設計記述に適用する上で、設計記述解析のための共通データ構造を提案し、前章までの提案手法を実装してその有効性を示している。従来からソフトウェア記述解析のためのデータ構造としてプログラムスライシングの分野で System Dependence Graph（SDG）が提案されており、デバッグなど各種解析に利用されている。ハードウェア高位設計記述を表現するには、SDG に対し、並列性やその同期メカニズム、ビットベクターなどのデータ構造、通信チャネルなども表現する必要がある。また、記号シミュレーションなどを実行するためには、因果関係のみでなく元の記述の情報も必要となるため Abstract Syntax Tree（AST）も必要となる。ExSDG はこれらを１つのデータ構造に融合したいもので、高位設計記述から ExSDG を自動生成する手法が示されている。

第７章は、Conclusion and Future Work（結論と今後の課題）と題し、本論文の研究成果をまとめるとともに、今後の発展方向について議論している。

以上、SoC や組込み機器設計に対する高位設計において、設計の正しさを数学的に証明する形式的検証技術に関し、設計対象の動作を制御部分と演算部分（データパス部分）に自動的に分け、それぞれを効率的に解析する新規手法を考案し実際にツール化し評価することで、実用規模のハードウェア・ソフトウェア協調設計の形式的検証が実現可能であることを実証しており、電子工学発展に寄与する点は少なくない。

よって本論文は博士（工学）の学位請求論文として合格したものを認められる。