IPA

2023 年度 未踏 IT 人材発掘・育成事業 採択案件評価書

1. 担当 PM

田中 邦裕（さくらインターネット株式会社 代表取締役社長）

2. クリエータ氏名

中神 悠太（筑波大学 情報学群 情報メディア創成学類）

3. 委託金支払額

2,692,300 円

4. テーマ名

RISC-V ベースのプロセッサを自動生成するシステムの開発

5. 関連 Web サイト

Sasanqua：https://github.com/SasanquaProject

6. テーマ概要

本プロジェクトは、FPGA を用いた開発の難易度を低減し、より多くの開発者による利用を促進することを目的としている。FPGA はその柔軟性から多様な応用が期待される一方で、ハードウェアに関する深い知識が必要とされるなど、開発のハードルは高い。本プロジェクトでは、この問題に対処するために、RISC-V ベースのプロセッサコアおよびそれを拡張するコプロセッサを開発し、これらを組み合わせて利用するためのプロセッサ生成ツールを提供する。

プロセッサコアは、RISC-V の基本命令セットである RV32I を実装しており、汎用性の高いプロセッシングを可能とする。一方、コプロセッサは、プロセッサコアには含まれない拡張命令を提供し、特定の処理を効率的に実行するための機能を担う。これにより、開発者は用途に応じてプロセッサの機能を拡張することが可能となる。

さらに、プロセッサ生成ツールを用いることで、プロセッサコアとコプロセッサを組み合わせ、開発者自身のニーズに合わせたカスタマイズが容易に行える。このツールは、開発者がプロセッサの構成を自由に設計し、FPGA 上で実装するためのサポートを提供する。

7. 採択理由

本プロジェクトでは、FPGA を使うにあたっての学習コストや設計の難しさなどを低減させ、簡単に FPGA の恩恵を受けられる開発ソフトウェアと制御ソフトウェアを開発するものである。 FPGA は、そのデバイスの規模の範囲内で自由に回路を組み替えることができ、音声処理や画像処理などの分野において、CPU よりも高い性能を発揮することができるが、回路の構築や設計などに対する高いスキルを必要とするため、利用のハードルが高い。そのため本プロジェクトでは、FPGA 上に RISC-V をベースとしたプロセッサをコアとして、そこにユーザに応じたコプロセッサを自由に配置し、CPU では実現できないような機能性を持つプロセッサを簡単に生成できるようにすることで、ユーザが必要とする機能を実現しようとする。

すでに同様の先行研究はあるが、今回のアプローチのように個人開発者を対象として、簡単にプロセッサの生成を行うというアプローチは新しく、未踏性があるものと考えて採択した。

8. 開発目標

本プロジェクトは、FPGA を対象としたプロセッサの自動生成システムの開発し、FPGA を活用するためのプロセスを容易かつシンプルになることを目標としている。具体的には、RISC-V を ISA に採用したプロセッサコアの開発、ユーザが自由に組み合わせ可能なコプロセッサ群の開発、プロセッサコア上で動作する軽量な専用制御ソフトウェアの開発、およびこれらの回路群と制御ソフトウェアを管理し、検証・実装を行うためのソフトウェアの開発を行う。RISC-V の拡張性を利用し、ユーザが自由に機能を選択してプロセッサを自動生成できるシステムを構築することが本プロジェクトの核心であり、プロセッサコアには基本命令のみを実装し、拡張命令をそれぞれ実装したコプロセッサ群としてプロセッサコアに対して結合可能にする。これらにより、ユーザは目的に応じて必要な命令を組み合わせてプロセッサを実装することが可能となり、FPGA 開発の敷居を大幅に下げることを目標とする。

- RISC-V ベースのプロセッサコアの開発
  パイプラインや分岐予測など、一般的な高速化手法を標準で実装し提供する。コプロセッサを組み合わせることによって機能を拡張できる。
- プロセッサコアに対して、ユーザが自由に組み合わせ可能なコプロセッサ群の開発
  ユーザが使用したい命令を持つコプロセッサを自由に選択し、プロセッサコアに結合することができる。ユーザが独自に構成する回路をラップする形で、ユーザは自由に独自コプロセッサを開発できる。
- プロセッサコア上で動作する軽量な専用制御ソフトウェアの開発

外部との通信、ユーザアプリケーションの管理、コプロセッサとの連携機能などを実装する。

- 上記の回路群・制御ソフトウェアを管理し、検証・実装を行うためのソフトウェアの開発

  ユーザはこのソフトウェアを使用することでプロセッサのカスタマイズ等を行う。論理合成や配置配線といった FPGA 開発特有の作業を簡単に行えるようにする。

加えて、本プロジェクトは、提案システムを実現した上で、実用的なプロセッサを 2 つ以上実際に生成し公開することを目指し、成果をオープンソースプロジェクトとして公開し、RISC-V コミュニティへの貢献を目指す。

9. 進捗概要

本プロジェクトでは FPGA 開発に関する開発を容易に行えるようになることを目指し、「選択」「合成」「実装」の 3 つのステップを経て、FPGA 上で動作する RISC-V プロセッサを生成することができる環境を作る。具体的には、図 1 に示される通り FPGA 上で動作するプロセッサの核となる「プロセッサコア」、プロセッサコアを拡張するための回路である「コプロセッサ」、プロセッサコアとコプロセッサを適切に合成するための「プロセッサ生成ツール」を開発する。また、これらを扱うための「ツールキット」を開発する。そして、これらをオープンソースとして公開することで、FPGA 開発を行いたい、あるいは行っている開発者に対して広く利用されることを目指した。

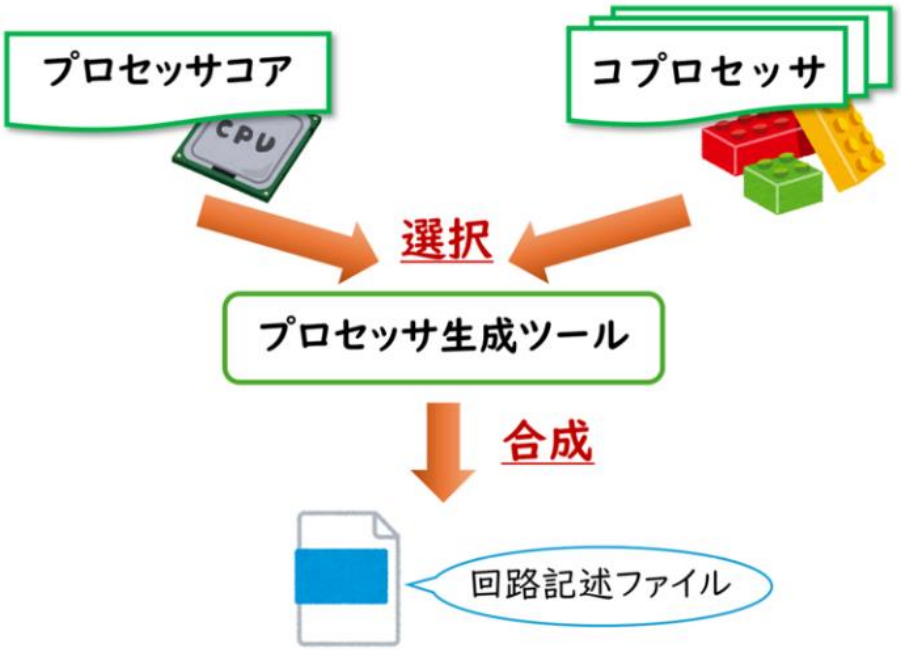

図 1：開発する回路およびソフトウェアの関係性

本プロジェクトでは独自に「Logic as Software」なる語を作成した。これは既存の語である IaaS や PaaS を基に作成した語であり、回路をまるでソフトウェアのように扱うことができるという意味を持つ。ソフトウェアの特性である再利用性や拡張性、変更の容易さを回路にも持ち込むことを目指し、本プロジェクトでは、回路およびソフトウェアの開発を行う。

本プロジェクトでは実際に、その成果としてプロセッサコア、コプロセッサ、プロセッサ生成ツールを開発し、GitHub 上で OSS として公開した。本プロジェクトで開発したプロセッサは FPGA 実機で動作し、50〜80MHz 程度のクロック周波数で動作する。また、ツールキットについても完全ではないが開発を行い、シンプルな GUI を持つソフトウェアを開発した。これについても GitHub 上で OSS として公開した。

本プロジェクトの開発は、Digilent 社が開発および提供をする ZYNQ Zybo-Z710 評価ボードを用いて行った。ZYNQ Zybo-Z710 評価ボードは、Xilinx 社の ZYNQ-7000 シリーズを搭載した評価ボードであり、FPGA と ARM プロセッサを 1 つのチップに搭載するものである。ZYNQ SoC を用いた開発では、FPGA と ARM プロセッサを連携させることができるため、FPGA 上で動作するプロセッサを開発するための環境として適している。また、ZYNQ Zybo-Z710 評価ボードは、多くの I/O ピンを持つため様々なデバイスとの接続が容易であり、検証環境としても適していることから、本プロジェクトではこの評価ボードを用いて開発を行った。

ZYNQ SoC を使用することから、回路の合成や実装には Xilinx 社の Vivado というツールを使用した。Vivado は FPGA の回路設計から合成、実装、およびプログラムの書き込みまでを行うツールであり、FPGA 開発において広く使用されているものである。併せて、Vivado に付属する Vitis IDE というツールを使用して、ZYNQ に含まれる ARM プロセッサのプログラム開発を行った。

プロセッサコアは ISA として RISC-V を採用し、基本命令として RV32I を実装した。RISC-V には様々な拡張命令が存在するが、本プロジェクトでは基本命令のみを実装した。プロセッサコアは、RV32I の基本命令を実装することで汎用的なプロセッサとしての機能の実装のみに注力させ、拡張命令は後述するコプロセッサで実装することとした。このようにすることで、プロセッサコアの設計をシンプルにすることができる。また、シンプルにすることで設計および実装工程でのバグ混入を防止する意図がある。

プロセッサコアの実装は HDL の一種である Verilog HDL を用いて行った。プロセッサコアの実装にあたっては設計図をもとに、各モジュールを実装した。モジュール性を重視した設計に基づき各モジュールのインタフェースを統一することで、実際に各モジュールの実装負荷を下げることができた。各モジュールの実装はそれぞれファイル分割しており、ソースコードの行数にしてそれぞれ約 300 行程度の規模となっている。全体としては約 5000 行程度であり、設計図

に示したモジュールの数に対して適切な規模であると言える。

コプロセッサは、プロセッサコアが持つ RV32I の基本命令に対して拡張命令を提供するためのモジュールである。コプロセッサはプロセッサコアと同様に、RISC-V の ISA に基づくものとし、この制限において自由に拡張命令を実装できるようにした。また、RISC-V は独自命令用にカスタム命令フィールドを持つため、このフィールドを使用して拡張命令を実装することができる。

コプロセッサの実装はプロセッサコアの実装と同様に、Verilog HDL を用いて行った。コプロセッサは具体的な実装をほとんど持たず、プロセッサコアとのインタフェースを持つモジュール（以下、Cop モジュール）として実装している。コプロセッサが持つ具体的な実装は、プロセッサコアとやり取りをする部分のみであり、それ以外の部分はコプロセッサの開発者が実装することを想定している。

プロセッサコアおよびコプロセッサは接続のための専用インタフェースを持つため、プロセッサ生成ツールはこれを適切に接続するように回路を生成する。このとき、プロセッサコアとコプロセッサを単にワイヤを用いて接続するだけではなく、間に「調停回路」という専用の回路を挟むことで、プロセッサコアとコプロセッサの間での独自の通信制御を行うようにする。調停回路はプロセッサコアからの信号を受け取り、これを適切にコプロセッサに伝え、またその逆の働きもする。例えば、プロセッサコアに複数のコプロセッサが接続されている場合、プロセッサコアから渡される命令を分配する必要があるほか、コプロセッサの応答を適切に選択してプロセッサコアに返す必要がある。これらの制御を調停回路が担当する。

このほか、本プロジェクトで開発したプロセッサコアやコプロセッサ、プロセッサツールを容易に扱うためのソフトウェアとして、GUI を持つツールキットの開発に着手した。ツールキットはユーザに対して、プロセッサコアやコプロセッサの組み合わせ指示、プロセッサ生成ツールの実行、生成された回路の合成および配置配線、FPGA への書き込みといった一連の作業を GUI で提供することを目指している。ただし、スケジュール上の都合により、ツールキットの開発は完了していない。

本プロジェクトで開発したプロセッサコアやコプロセッサ、プロセッサ生成ツールは、全て GitHub 上で公開している。公開にあたっては GitHub Organization を作成し、そこにリポジトリをまとめて公開している。それぞれの成果物は次に示すリポジトリで公開している。

- https://github.com/SasanquaProject/Sasanqua：メインリポジトリ
- https://github.com/SasanquaProject/Core-IP：プロセッサコアの実装
- https://github.com/SasanquaProject/ToolKit：ツールキット

以上の通り、RISC-V の基本命令セット RV32I を実装したプロセッサコアおよび、プロセッサコアの機能拡張を可能にするコプロセッサが開発された。これにより、開発者は FPGA 開発の初期段階でのハードルが低減され、特定の処理を高速化するためのカスタマイズが容易になった。また、プロセッサコアとコプロセッサを組み合わせて、独自のプロセッサを自動生成できるソフトウェアツールが提供され、開発者はプログラミング言語による記述でプロセッサを設計できるようになった。

これらの成果により、本プロジェクトは FPGA 開発のアクセシビリティを向上させるとともに、RISC-V コミュニティの発展に貢献している。

10. プロジェクト評価

本プロジェクトが目的としていた「FPGA 開発の難易度を下げる」については、プロセッサコアやコプロセッサ、およびこれらを扱うためのツールキットを開発し、FPGA 開発における革新的な手法を提案しており、達成されているものと考えられる。本プロジェクトのコンセプトである「Software as Logic」は、従来のプログラミング言語に慣れ親しんだ開発者にとって理解しやすく、FPGA 開発の入門障壁を著しく低減するものである。特に、HDL と Rust プログラムを組み合わせることによるハードウェアレイヤの隠蔽は、開発者がより高いレベルでの設計と制御に集中できるようにし、FPGA 開発の効率化と簡素化に大きく貢献していると言っても過言ではない。

また、本プロジェクトにおける HDL による回路設計の補助は、FPGA 開発に必要不可欠なスキルであるにもかかわらず、その複雑さから開発のハードルを高めている問題を解決するものである。プロセッサ生成ツールを用いることで、開発者は必要とされる Verilog プログラムの量を大幅に削減でき、プロセッサコアが動作に必要なほとんどを担っているため、記述するプログラムの難易度も低減される。このようなアプローチにより、FPGA 開発における自由度を高めつつ、開発工程を短縮することが可能となる。

11. 今後の課題

本プロジェクトを通じて、FPGA 開発のハードルをさらに下げることには貢献できた。ただし、プロセッサコアの拡張の面では、現在実装されている RV32I 命令セットに加えて、RV32M, RV32C, RV32A などの追加命令セットの実装が必要とされている。それは Linux の動作を目指す場合に必須の拡張であり、現在の設計では、RV32C の圧縮命令セットや RV32A のアトミック命令の実装には困難が伴うため、プロセッサコアの設計と実装の発展が求められる。

また、並列実行に関しては、現在のプロセッサコアは簡易的な並列動作が可能であるものの、命令単位での並列実行が実現できていない。今後の課題として、命令単位での並列実行を実装し、より高度で実用的なプロセッサコアの実現を

目指すことが挙げられる。

コプロセッサの拡張では、異なる HDL の採用やパッケージ管理システムの導入が検討されている。これにより、コプロセッサの開発が容易になり、共有や利用がより便利になることが期待される。

ツールキットの発展については、現在の機能が限定的であるため、GUI の改善や FPGA の実装を完全にツールキット上で行えるようにするなど、今後の発展が求められる。これらの取り組みにより、プロジェクトのコンセプトである「Logic as Software」を実現し、FPGA 開発のハードルをさらに下げることが目指されている。

本プロジェクトは、FPGA 開発のハードルを下げることを目的としており、プロセッサコアとコプロセッサの設計、実装、およびこれらを統合するための専用ソフトウェアの開発に取り組んでいる。さらに、これらを扱うためのツールキットの開発も進行中である。

以上の通り、プロジェクトの初期目標である「FPGA 開発のハードルを下げる」という点をさらに進めるために、さらなる開発が期待される。