RISC-V Days Tokyo 2022 Spring

imperas

ImperasDVおよび新たなオープンスタンダードRVVIを利用した高品質なRISC-V向け検証環境

Shuzo Tanaka, eSOL TRINITY Co., Ltd.

Co-Author: Simon Davidmann & Lee Moore – Imperas Software

31 May 2022

議題

- Imperas社紹介
- RISC-V processor 設計検証（DV:Design Verification)の課題
- 5レベルのRISC-V 設計検証手法
- DVにおける重要な技術: リファレンス モデル, 検証用 IP
- まとめ

議題

- Imperas社紹介
- RISC-V processor 設計検証（DV:Design Verification)の課題
- 5レベルのRISC-V 設計検証方法
- DVにおける重要な技術: リファレンス モデル, 検証用 IP
- まとめ

Imperas会社紹介

- Imperasを設立したチームのメンバーは、EDAツール、FPGAおよびプロセッサIPに関する技術バックグラウンドを持っています。
- 彼らは、EDAツールのようなシミュレーション手法が、今後のソフトウェア開発、テスト等で必須になるであろうと考え、Imperas社を設立しました。
- シミュレーション無しではSoCは開発できない！このように組込みシステム開発においてもシミュレーション無しでは開発および出荷が出来ないと考えます。
- Imperas社製品の特長: Imperas製品は、ツールへの技術要求から製品アーキテクチャを検討し、開発されています。モデリング要件からのボトムアップで開発されたツールではありません。
- Imperas社は、RISC-Vの普及により、新たにRTLの設計検証環境の技術および手法をポートフォリオに加えました。

Imperas社のRISC-Vへの取組み

- Q2 2016: DAC – 最初のRISC-Vとの出会い – まだアカデミックな状況と認識
- Q4 2016: RISC-V Workshop (@ Google) – 350社が参加、ISAは収束しつつある状況
- Q1 2017: ImperasがRISC-V Foundationに参加; 最初のRISC-V processor modelを開発
- Q3 2017: ImperasがCompliance Working Groupに参加; ISSの構築・寄贈とテストを実施
- Q1 2018: ImperasがRISC-Vコアへの命令追加、最適化の手法を導入
- Q2 2018: ソフトウェア開発および設計検証用途にImperasのRISC-V modelsを初めて購入して頂いた
- Q1 2019: ImperasのRISC-Vモデルを設計検証モデルとして使用したRISC-V SoCの最初のテープアウト
- Q2 2019: ImperasとGoogleが、命令ストリームジェネレータを使用したDVフローで協業を開始
- Q1 2020: Imperasは、OpenHW Groupおよび各メンバーとCore-VコアのDVに関する研究を開始
- Q1 2021: Imperasは、DVConシリコンバレーカンファレンスにて、RISC-VのDVに関する論文を2本発表 (OpenHW、Nvidia Networkingと共同)
- Q4 2021: Imperasは、ImperasDV RISC-V向け検証製品を製品ラインナップとして発表

Imperas社のRISC-V関連のお客様およびパートナー

The most complex RISC-V processor projects use Imperas

Users

- Nagravision
- Nvidia Networking (Mellanox)
- EM Micro US
- Silicon Labs
- Dolphin Design
- lowRISC (Ibex)
- RISC-V processor IP vendor
- Top-tier systems company (AI application)
- Top-tier consumer electronics company (AR/VR application)
- **NSITEXE (DENSO subsidiary)**
- Startup building accelerator based on multiprocessor RV64
- **Japanese government projects “TRASIO” and “RVSPF”**
- Barcelona Supercomputing Center
- Numerous universities around the world

Partners

- RISC-V Intl (compliance, bitmanip, crypto, various events)
- OpenHW (verification working group)
- CHIPS Alliance (verification working group)
- Google (for open source ISG, & co-author DV papers)
- Valtrix (test generation tools)
- Andes (processor IP vendor)
- SiFive (processor IP vendor)
- Codasip (processor IP vendor)
- MIPS (processor IP vendor)

議題

- Imperas社紹介
- RISC-V processor 設計検証（DV:Design Verification)の課題
- 5レベルのRISC-V 設計検証方法
- DVにおける重要な技術: リファレンス モデル, 検証用 IP
- まとめ

RISC-V Processor設計検証の課題

- IPの機能選択は、設計および実装に大きく影響します。
  - 経験のあるプロセッサ開発チームは、各機能に掛かるコストについての知見を持っています。
    - 機能追加の度に検証の複雑度が増していきます。
    - 単純な機能追加においても大きなコスト増になりえます。
    - スケジュールの遅延、追加リソースの投入等が必要になり、追加における品質コスト増が大きなリスクになります。
- 2021年の時点でプロセッサDVツールは、製品として販売されていませんでした
  - EDAベンダーで‘RISC-V CPU DV kit’製品は存在しませんでした
  - CPUベンダー(例えば、Intel, AMD, Arm等)は自社内のソリューションとして環境を構築していましたが、販売はしていません。
  - プロセッサを自社で開発するリスク・・・開発スケジュール、人員、品質に掛かるコスト等
- 現在のSoC開発に掛かるコストの50%はハードウェアDVです (実績のあるCPU IPを購入した場合)
  - 自社のCPUを開発するには、多くのDVコストを見込んでおく必要があります。

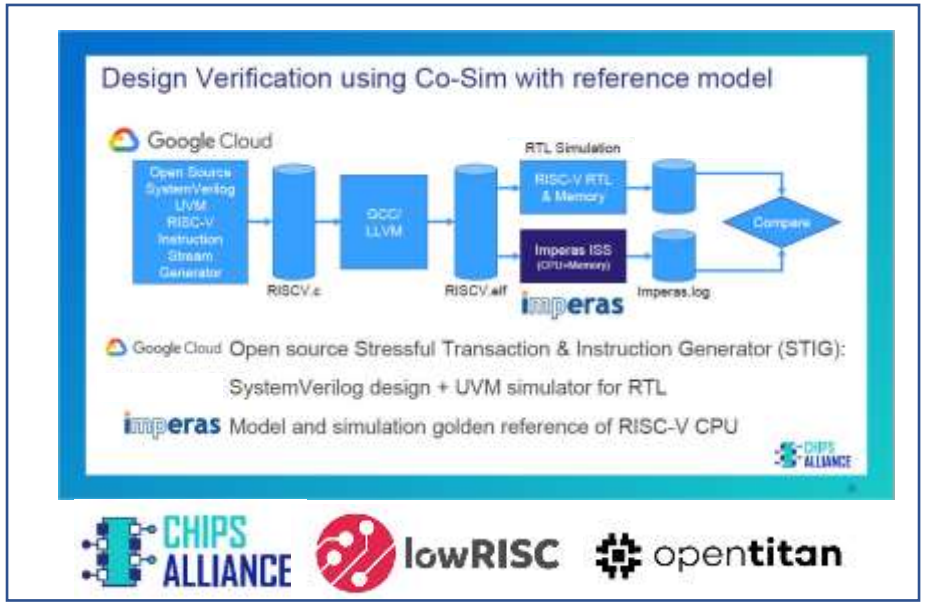
議題

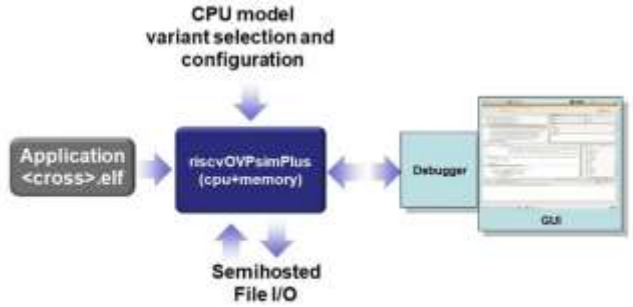
- Imperas社紹介
- RISC-V processor 設計検証（DV:Design Verification)の課題
- **5レベルのRISC-V 設計検証方法**
  1) Hello World
  2) Self-checking tests (e.g. Berkeley torture tests pre-2018)
  3) **Post-simulation trace log file compare**
  4) 同期 step-and-compare
  5) **非同期 step-and-compare**
- Key technologies: reference models, verification IP
- Summary

ここは割愛

ここは割愛

3) Post-Simulation Trace Log File Compare (Entry Level DV)





Imperas riscvOVPsimPlus Reference Simulator

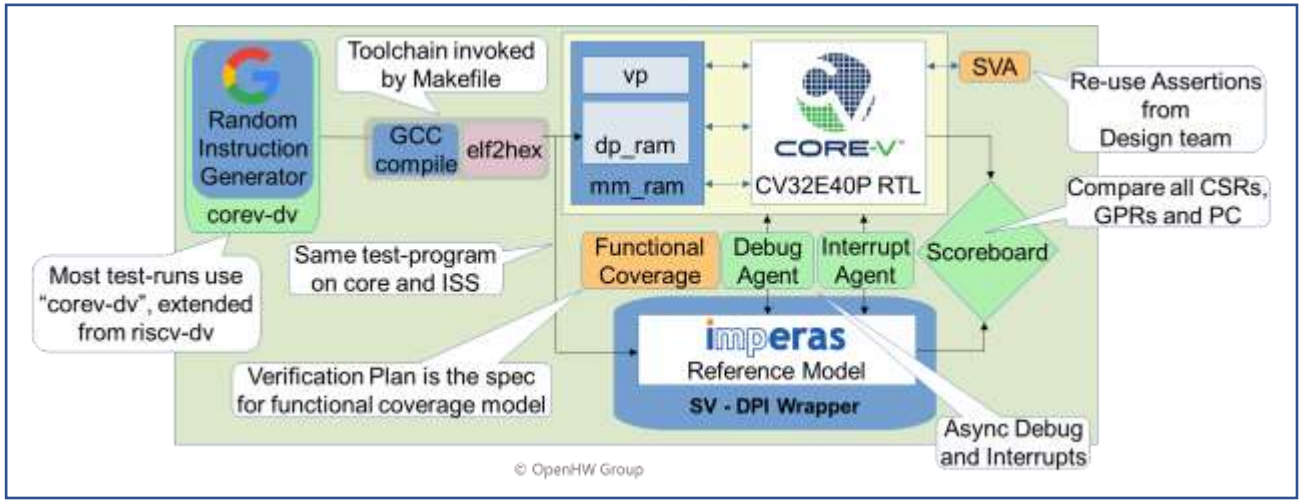
- Process
  - ランダム命令生成ツール (ISG)を使用しテストケースを作成
  - ISSがシミュレーション中に実行ログを出力
  - RTLシミュレーション中に実行ログを出力
  - 両シミュレーションの終了後両方のログを比較し違いを確認する
- ISS: riscvOVPsimPlus（トレースおよびGDBインタフェースを持つ）
  - Free ISS: https://www.ovpworld.org/riscvOVPsimPlus
- ISG: riscv-dv from Google Cloud / Chips Alliance
  - Free ISG: https://github.com/google/riscv-dv

5) 非同期 Step-Compare (高品質 DV 手法)

- 検証環境には、以下のような複雑な機能を検証する為の機能が必要とされます。アウトオブオーダー、マルチパイプライン、マルチハート、デバッグモード、割込み等 …
  - 例えば、SystemVerilogのコンポーネントでは、以下の機能が用意されています。
    - トレーサー: 命令のチェックとレジスタへの書込みレポート機能
    - ステップ実行と比較: リファレンスモデルの制御と機能のチェック
    - 割込み入力: 割込みのカバレッジとチェック
    - デバッグ: デバッグ機能のカバレッジとチェック

- 一般的に困難で複雑、コストが掛かる作業です
  - 課題は、マイクロアーキテクチャRTLパイプラインから非同期情報を取得することです。

Example flow:



2nd generation CV32E40P OpenHW flow (2H2020)
(Imperas model encapsulated in SystemVerilog)

非同期Step-Compare Summary

- **非同期step-compare方式は包括的で最も効率的なDVアプローチと考えます。**
  - 命令単位でのロックステップ実行および比較 (非同期イベントを含む)を行う
    - 実行フローの比較、プログラムデータの比較、および内部ステートの比較等
  - 命令単位で実行しながら結果の比較を実行
    - エラーが検出された時点からのデバッグが可能。効率的なデバッグが可能
    - エラー検出後の無駄な実行を行わない。無駄なシミュレーション時間を使わない。
  - 制御フローやハードウェア実行時間にも影響する非同期イベントの検証が可能
  - マルチハートプロセッサ、アウトオブオーダー、マルチパイプライン等の複雑な設計にも対応

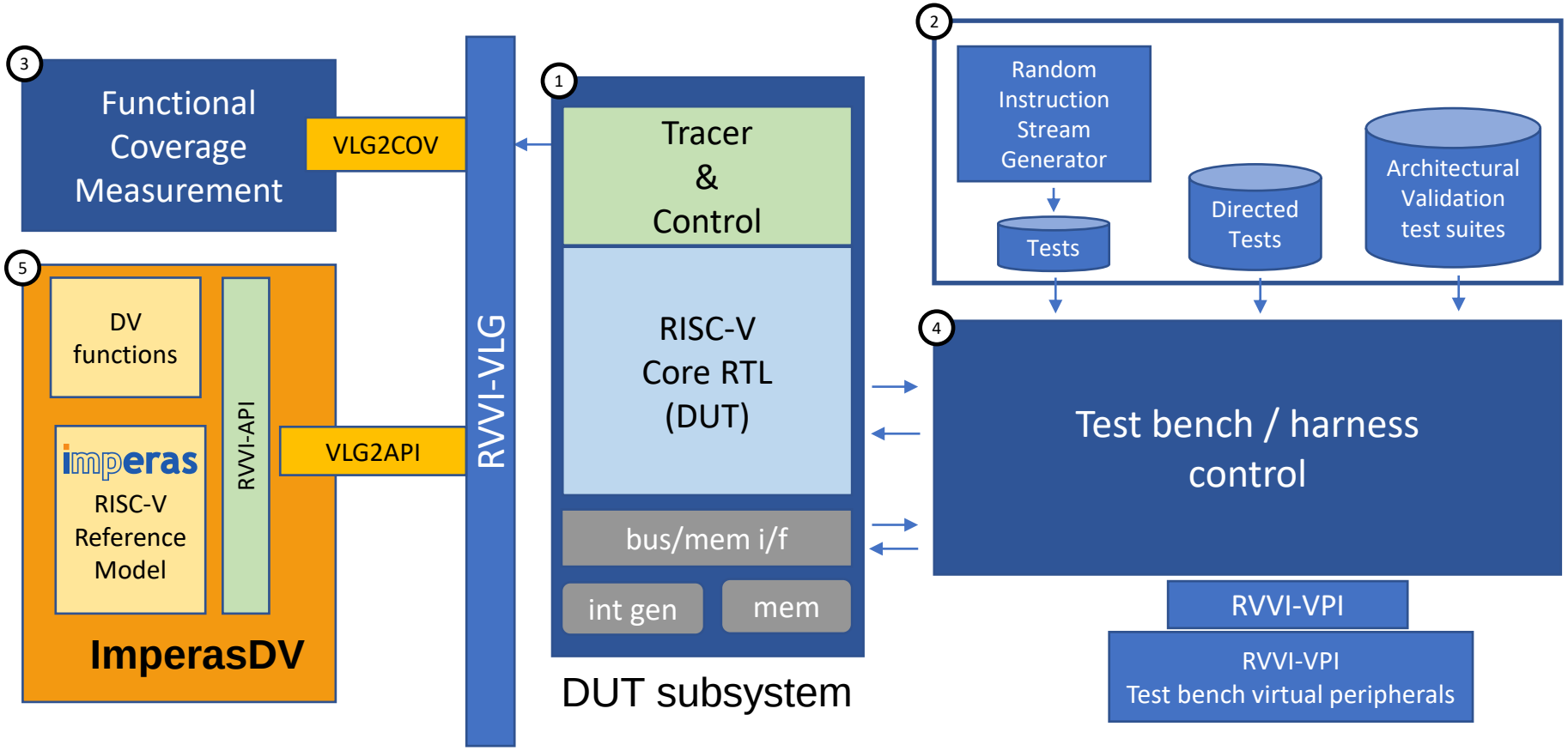
- **課題**
  - 環境開発および設定が難しい (RTL DUT tracerの機能とパイプライン構造の理解に依存)
  - バグが発見された時にコストが掛かる。時間、リソース、ツールライセンス費用が必要
  - **しかし、バグは早期に見つけないとさらにコストが掛かることに・・・・**

- **非同期step-compareの次のステップ: テストベンチの標準化; 検証用IPの開発**

議題

- Imperas社紹介
- RISC-V processor 設計検証（DV:Design Verification)の課題
- 5レベルのRISC-V 設計検証方法
- DVにおける重要な技術: リファレンス モデル, 検証用 IP
- まとめ

RVVIを利用したRISC-Vプロセッサ DV環境



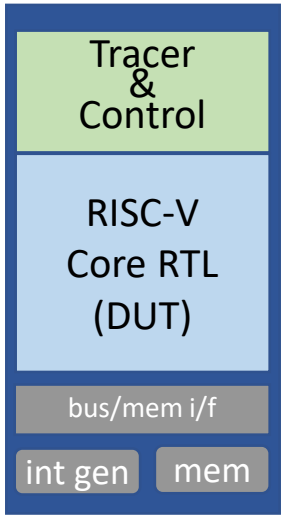
5 components of RISC-V processor hardware DV

1. DUT subsystem with 'tracer'
2. Tests: (random) instruction test generator and directed tests
3. Functional coverage measurement
4. Test bench / harness
5. DV subsystem

ImperasDV
https://www.imperas.com/imperasdv

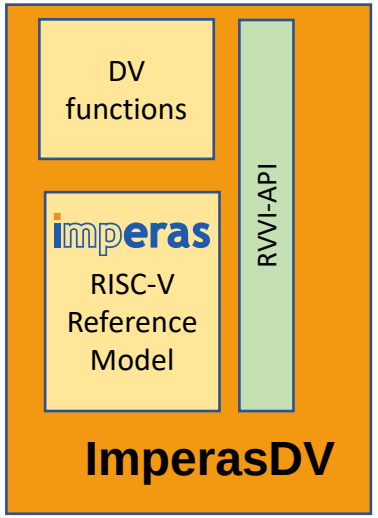
RVVI: RISC-V verification Interface
https://github.com/riscv-verification/RVVI

RVVI: 標準化された、テストハーネスの結線に関するRISC-V検証インタフェース



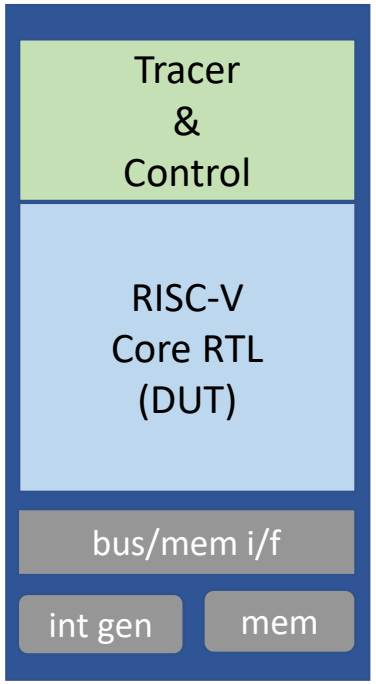
DUT subsystem

- https://github.com/riscv-verification/RVVI (Public Open Standard)
- RVVI-VLG
  - Verilog DUT interfaces
    - RVVI-VLG state – ストリーミング 'tracer' データ
    - RVVI-VLG nets – 実装依存 (割込み、デバッグ)
    - マルチハート、マルチパイプライン、アウトオブオーダーのサポート



- RVVI-API
  - DV subsystemとリファレンスモデルの制御
  - RVVI_state - RISC-V Verification Interface - State
  - RVVI_control - RISC-V Verification Interface - Control
  - RVVI_io - RISC-V Verification Interface - IO (Interrupts, Debug)
  - RVVI_bus - RISC-V Verification Interface - (Data, Instruction Bus)
  - SystemVerilog、C、C++テストベンチのサポート

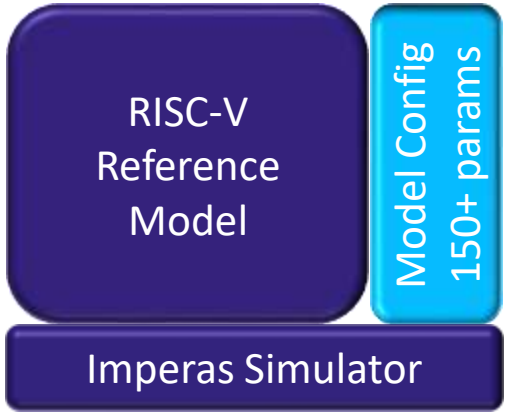
RTL DUTのトレースインタフェース



DUT subsystem

- DUTをテストする上で重要なコンポーネント
  - メモリモデルとバスインタフェースを含む
  - 割込み発生機能を含む
- トレーサーには適当なデータをテストベンチに提供する機能が要求されます。
- 制御インタフェースには、テストベンチがイベントをスルー出来る機能が要求されます。
- **トレーサーの持つ機能がDV環境の機能に大きく影響します！**

Imperasのリファレンスモデル

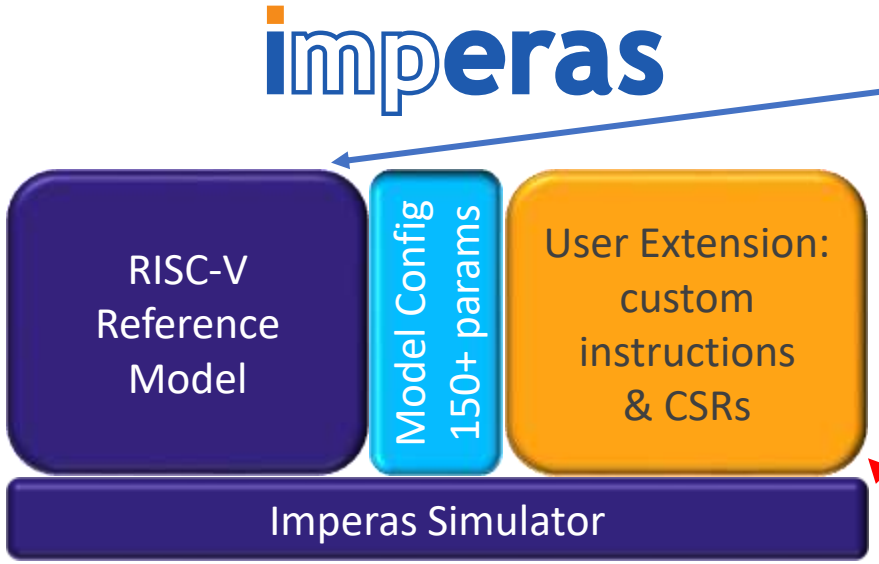


http://www.imperas.com/riscv

- ImperasはRISC-Vフル仕様のモデルを提供します。
- 製品品質のRISC-Vプロセッサモデル/シミュレータはコンプライアンス、検証およびテスト環境として使用されています。
- 全ての機能が実装されたコンフィギュレーション可能なモデル / シミュレータ
  - ユーザーおよび特権モード仕様をサポートした全ての32bitおよび64bit機能
  - ベクター拡張命令：versions 0.7.1, 0.8, 0.9, 1.0をサポート
  - ビット操作拡張命令：versions 0.91, 0.92, 0.93, 1.0.0をサポート
  - ハイパーバイザー：version 0.6.1をサポート
  - K-Cryptoスカラー：version 0.7.1, 1.0.0をサポート
  - デバッグ：versions 0.13.2, 0.14, 1.0.0をサポート
- モデルのソースコードはApache 2.0 open sourceライセンスで公開
- リファレンスモデルとして以下のお客様で使用されています。
  - Mellanox/Nvidia, Seagate, NSITEXE/Denso, Google Cloud, Chips Alliance, lowRISC, OpenHW Group, Andes, Valtrix, SiFive, Codasip, MIPS, Nagra/Kudelski, Silicon Labs, RISC-V Compliance Working Group, …

ImperasのモデルはRISC-V Goldenリファレンスモデルとして利用されています

Imperasモデルの拡張性について



- 分割され、重複の無いソースファイルがメンテナンスを容易にします
- Imperasまたはお客様が拡張部分を開発可能です
- ユーザー拡張のソースコードはお客様の権利となります

Imperasが開発およびメンテナンスを行う基本モデル

- 基本モデルは全てのRISC-V仕様が実装されています
- どの拡張ISAを使用するかを自由にコンフィギュレーションできます。
- また、拡張ISAの中のどのバージョンの拡張命令を使用するかをコンフィギュレーションできます。
  - ISA拡張命令仕様のアップデートに合わせてモデルも更新されます。
- RISC-Vのオプション仕様のコンフィギュレーションが可能です
  - 例えば、オプションのCSRs, read onlyまたはread/writeビットオプション等

Imperasは拡張可能な基本モデルの拡張方法を提供します

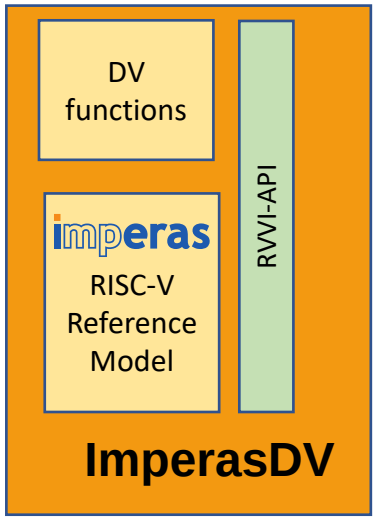
- カスタム命令追加の為のテンプレート
- 機能追加の為のコード
- ユーザーガイド/多くのサンプルを含むリファレンスマニュアル
  - プロセッサモデルの拡張サンプル等

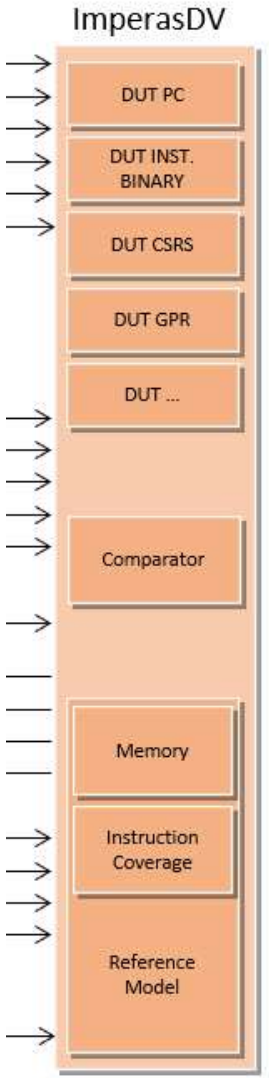
Imperasのモデルは容易に拡張、およびメンテナンスが可能な
ような構造に設計されています

ImperasDV: RISC-V 検証用 IP

- 検証用IPは以下の為に必要となります
  - 使いやすさ
  - スケーラビリティ
  - 拡張性
  - 性能
  - デバッグ
  - 工数削減

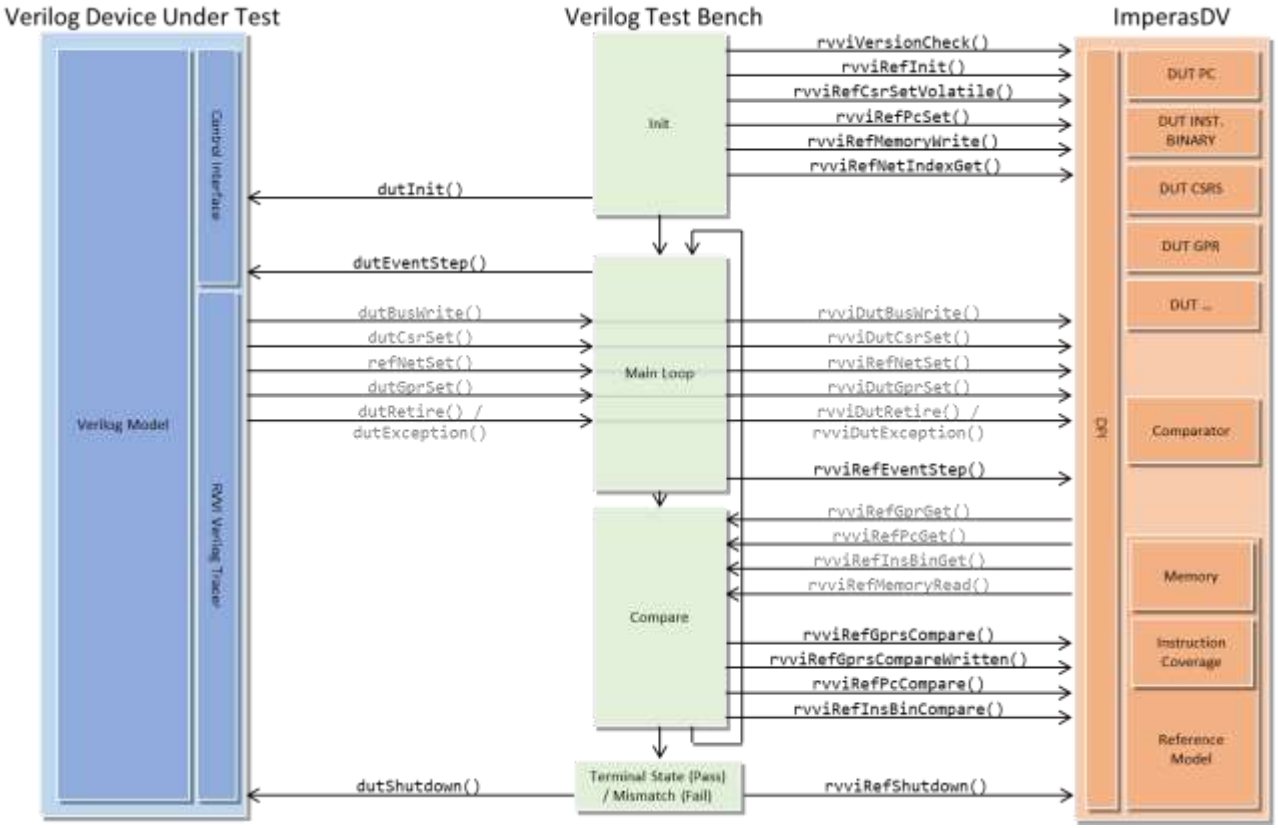
DV 機能





- モデル選択、バージョン、コンフィギュレーション
- リファレンスモデルのカプセル化
  - 命令カバレッジの計測
- DUT参照ステートの保管
- 同期技術
  - 同期および非同期実行、割込み、デバッグ、マルチハート
- 比較技術
  - 命令実行後の比較；マルチパイプラインおよびアウトオブオーダープロセッサ用のDV
- C/C++ または SystemVerilogテストベンチ / ハーネス
  - RVVI-APIの利用
- シンプルな使い方 – the ‘smarts’ are built-in

ImperasDVのセットアップ



- リファレンスモデルの設定
- レジスタおよびメモリの初期化コンフィギュレーション
- 比較対象を選択 (DUTトレーサーの機能に依存します)
  - PC, GPR, CSR, FPR, VR, decode, net, hart …
- 方式を選択:
  - 同期step-compareまたは非同期step-compare
- トレースおよびログ取得の設定
- 命令カバレッジの選択
- DV制御オプションの選択

議題

- Imperas社紹介
- RISC-V processor 設計検証（DV:Design Verification)の課題
- 5レベルのRISC-V 設計検証方法
- DVにおける重要な技術: リファレンス モデル, 検証用 IP
- まとめ

まとめ

- RISC-Vプロセッサの開発者は、RISC-Vの柔軟な仕様に対応する為の高品質なRTL検証環境が必要です。
- プロセッサDV手法はImperasとお客様、およびパートナーとで進化させてきました。その成果はオープンスタンダードなRVVI に繋がっています。
- 非同期*lock-step-compare*手法は包括的で効率的および適合にすぐれたRISC-V設計検証フローです。
- 重要な技術はImperas RISC-Vリファレンスモデルと ImperasDVに含まれる検証用IPになります。

imperas

Thank you!

Shuzo Tanaka, eSOL TRINITY Co., Ltd.

sh-tanaka@esol-trinity.co.jp

RVVI: RISC-V verification Interface
https://github.com/riscv-verification/RVVI

ImperasDV
https://www.imperas.com/imperasdv