Hybrid Simulation with Emulation for RISC-V Software Bring Up and Hardware-Software Co-Verification

Duncan Graham and Larry Lapides

June 2023
Agenda

• RISC-V: Who/What/Where/When/Why
• RISC-V: How ... to be successful
• Processor modeling
• Virtual and Hybrid Platforms with Helium, Palladium and Protium
• Summary
Agenda

• RISC-V: Who/What/Where/When/Why
• RISC-V: How ... to be successful
• Processor modeling
• Virtual and Hybrid Platforms with Helium, Palladium and Protium
• Summary
Imperas RISC-V Timeline

- **Q1 2017**: Imperas joins the RISC-V Foundation; builds first RISC-V processor model
- **Q3 2017**: Imperas starts participating in the Compliance Working Group; builds/donates tests
- **Q1 2018**: Imperas introduces methodology for adding/optimizing custom instructions (architecture exploration) for RISC-V cores
- **Q2 2018**: First paying customer using Imperas RISC-V models and tools for software development and design verification (DV)
- **Q4 2018**: First paying customer using Imperas RISC-V models and tools for architecture exploration
- **Q1 2019**: First tape out of RISC-V SoC based on using Imperas model as DV reference model
- **Q2 2019**: Imperas releases riscvOVPsimPlus, free instruction set simulator supporting the full RISC-V specification
- **Q1 2020**: Imperas starts working with the OpenHW Group and individual members on DV of Core-V cores
- **Q4 2021**: Imperas introduces ImperasDV RISC-V verification product line
- **Q1 2022**: Imperas introduces RVVI (RISC-V Verification Interface) as an open standard on GitHub for the RISC-V processor DV community
- **Q4 2022**: Imperas introduces functional coverage verification IP for RISC-V processor DV
- **Q2 2023**: Imperas and Cadence develop OEM relationship enabling Cadence to resell Imperas RISC-V models
RISC-V Freedom Enables Domain Specific Processing

- **Who:** RISC-V users include traditional semiconductor companies, and embedded systems companies now practicing vertical integration by developing their own SoCs
- **What:** RISC-V is an open instruction set architecture (ISA), it is not a processor implementation
- **Where:** RISC-V is growing in market segments where x86 (PCs, data centers) and Arm (mobile) architectures are not dominant
  - Small microcontrollers for SoC management, replacing proprietary cores
  - Verticals such as IoT and automotive
  - Horizontal markets such as security and AI/ML
  - Deep embedded applications
- **When:** RISC-V processors are now used in over 30% of SoCs
- **Why:** The freedom of the open ISA enables users to develop *differentiated* domain specific processors and processing systems
Agenda

- RISC-V: Who/What/Where/When/Why
- RISC-V: How ... to be successful
- Processor modeling
- Virtual and Hybrid Platforms with Helium, Palladium and Protium
- Summary
RISC-V Processor Complexity

- RISC-V is a modular instruction set architecture
- Any extension (functional group of instructions, e.g. atomics, compressed, floating point, vector) can be added to the base processor
- Then add in interrupts, privilege modes, Debug mode, multi-hart (multi-core), etc. and it gets complex
- Then processor DV, tool chain development and other software development is needed
RISC-V Processing Subsystems

- Multi-processor subsystems are commonly being developed using RISC-V cores
- Application areas include DSP, AI/ML and packet processing
- This adds complexity to both the DV and software development tasks
Technologies & Methodologies for Processor Verification and Software Development

- Processor verification needs RTL simulation for comprehensive processor DV
- Also needs an asynchronous-step-compare methodology, supported by verification IP and functional coverage
- Software development can use virtual platform simulation and FPGA prototypes
- These conventional techniques are not enough for the complex RISC-V processors and processing subsystems being designed today

- SoC verification, and software development and validation for the complex RISC-V processors and processing subsystems, requires hardware-software co-verification and prototyping using a hybrid software simulation – hardware emulation methodology
Example Hybrid Platform with Cadence Helium and Palladium

Functional Simulation

Hybrid Interface

RTL in Palladium/Protium
Agenda

• RISC-V: Who/What/Where/When/Why
• RISC-V: How ... to be successful
• Processor modeling
• Virtual and Hybrid Platforms with Helium, Palladium and Protium
• Summary
RISC-V Model Requirements

- Model the ISA, \textit{including all versions of the ratified spec}, and stable unratified extensions
- If the model is open source, users should not have to maintain their own fork of the model
- Model other behavioral components, e.g. interrupt controllers
- Easily update and configure the model(s) for the next project
- User-extendable for custom instructions, registers, ...
- Model actual processor IP, e.g. Andes, SiFive, Codasip, MIPS, NSITEXE, OpenHW, ...
- Well-defined test process including coverage metrics and mutation testing
- Interface to other simulators, e.g. SystemVerilog, SystemC, Imperas virtual platform simulators
- Interface to software debug tools, e.g. GDB/Eclipse, Imperas MPD
- Interface to software analysis tools including access to processor internal state, etc.
- Interface to architecture exploration tools including extensibility to timing estimation

- Most RISC-V ISSs can meet one or two of these requirements
- Imperas models and simulators were built to satisfy these requirements, and matured through usage on non-RISC-V ISAs over the last 15+ years
Imperas OVP RISC-V Models are used for Processor DV & SW Development and Architecture Analysis

- Base model implements RISC-V specification in full
- Fully user configurable to select which ISA extensions/versions
- Imperas provides methodology to easily extend base model

Processor DV Flow

Software Simulation Environment

Imperas DV

- Application Software & Operating System
- Virtual Platform
  - RISC-V Reference Model
  - RISC-V Core
  - RTL (DUT)
  - Testbench

- Simulation control
- UVM env (optional)
- SystemVerilog top level
- Testbench

- SystemVerilog
- C/C++

- Jitter simulator engine
- SlipStreamper API

- User Extension: custom instructions & CSRs
- Model Config 250+ params
- Vendor-specific config file & custom instructions

Imperas OVP RISC-V Models are used for Processor DV & SW Development and Architecture Analysis

- Base model implements RISC-V specification in full
- Fully user configurable to select which ISA extensions/versions
- Imperas provides methodology to easily extend base model

Processor DV Flow

Software Simulation Environment

Imperas DV

- Application Software & Operating System
- Virtual Platform
  - RISC-V Reference Model
  - RISC-V Core
  - RTL (DUT)
  - Testbench

- Simulation control
- UVM env (optional)
- SystemVerilog top level
- Testbench

- SystemVerilog
- C/C++

- Jitter simulator engine
- SlipStreamper API

- User Extension: custom instructions & CSRs
- Model Config 250+ params
- Vendor-specific config file & custom instructions

Imperas OVP RISC-V Models are used for Processor DV & SW Development and Architecture Analysis

- Base model implements RISC-V specification in full
- Fully user configurable to select which ISA extensions/versions
- Imperas provides methodology to easily extend base model

Processor DV Flow

Software Simulation Environment

Imperas DV

- Application Software & Operating System
- Virtual Platform
  - RISC-V Reference Model
  - RISC-V Core
  - RTL (DUT)
  - Testbench

- Simulation control
- UVM env (optional)
- SystemVerilog top level
- Testbench

- SystemVerilog
- C/C++

- Jitter simulator engine
- SlipStreamper API

- User Extension: custom instructions & CSRs
- Model Config 250+ params
- Vendor-specific config file & custom instructions

Imperas OVP RISC-V Models are used for Processor DV & SW Development and Architecture Analysis

- Base model implements RISC-V specification in full
- Fully user configurable to select which ISA extensions/versions
- Imperas provides methodology to easily extend base model

Processor DV Flow

Software Simulation Environment

Imperas DV

- Application Software & Operating System
- Virtual Platform
  - RISC-V Reference Model
  - RISC-V Core
  - RTL (DUT)
  - Testbench

- Simulation control
- UVM env (optional)
- SystemVerilog top level
- Testbench

- SystemVerilog
- C/C++

- Jitter simulator engine
- SlipStreamper API

- User Extension: custom instructions & CSRs
- Model Config 250+ params
- Vendor-specific config file & custom instructions

Imperas OVP RISC-V Models are used for Processor DV & SW Development and Architecture Analysis

- Base model implements RISC-V specification in full
- Fully user configurable to select which ISA extensions/versions
- Imperas provides methodology to easily extend base model
Models Drive Customization

- Custom instructions are added to optimize a specific application or set of applications within a domain

- Models let you explore quickly
  - Much faster to develop than RTL
  - Better profiling information available
  - Easier to debug software

- Methodology
  - Start by characterizing the application to be optimized
  - Then add the custom instructions, evaluate, and iterate
New custom instructions, new additional state registers

New custom instructions in trace disassembly
Imperas Models Easily Assembled into Virtual/Hybrid Platforms in Helium

Library of reference examples for virtual and hybrid platforms with bare-metal and Linux workloads to jump-start project development

Mix of Cadence, ARM, RISC-V models

Quickly configure transactors for connection of virtual components to RTL

Instantiate, configure and connect Imperas RISC-V models
Heterogeneous Multi-core Helium System-level Debugger

Heterogeneous Multi-core debugger can debug software running on ARM or RISC-V cores in same session, including complex workloads like Linux.

View and Debug Memories, IP registers throughout platform.

Integrated Disassembler from Imperas for custom RISC-V instruction display.

View Memory Map, Virtual/Physical Memory and Memory Mapped to RTL.

Integrated console for hardware or software debugging operations.
Agenda

• RISC-V: Who/What/Where/When/Why
• RISC-V: How ... to be successful
• Processor modeling
• Virtual and Hybrid Platforms with Helium, Palladium and Protium
• Summary
Why Hybrid Emulation-Simulation Systems?

- Hardware emulation is valuable for hardware-software co-verification, and for low level software development, porting and bring up
- Hardware emulation is expensive, typically a scarce resource in companies
- Hardware emulation is 100x slower than real time
  - Need to start execution from boot up or reset
  - What happens when the interesting events occur after billions of instructions, e.g. after Linux boot?
    - Can take minutes, or even hours, of “wasted” emulation time to get to the interesting events
- The hybrid system takes advantage of the speed of software simulation to get to the interesting events in seconds
Helium-Palladium SoC Hybrid Example

Embedded software debugger
Imperas-Cadence Joint Customer: Software Development/Optimization/Test on AI/ML SoC

- SoC has 140+ cores
  - Andes RISC-V cores with mix of scalar and vector processors
  - Processors used for a) AI/ML, b) running OS, c) SoC functions (power management, communications, ...)
- Platform software simulation runs @ > 500MIPS
- Users can run full platform, or subsets for AI/ML, FW, OS, ...
- Runs real software (production binaries)
- Software up and running in virtual platform one year before RTL tapeout
- Software ran within days on the first silicon
- **Helium Hybrid with Palladium emulation used for firmware testing**
Agenda

• RISC-V: Who/What/Where/When/Why
• RISC-V: How ... to be successful
• Processor modeling
• Hybrid simulation-emulation with Helium and Palladium
• Summary
High Quality RISC-V Models Are Required for RISC-V SoC Success

• Use cases
  • Processor and SoC verification
  • Software development, debug and test

• Imperas OVP Fast Processor Models satisfy RISC-V project requirements

• Hybrid simulation-emulation with Imperas RISC-V models is needed for complex RISC-V processors and/or RISC-V based SoCs
  • Connect RISC-V models to SoC RTL to co-develop software and hardware
Thank you

Duncan Graham
graham@imperas.com