SIM-V
Fast, Parallel RISC-V Simulation for Rapid Software Verification

Lukas Jünger, MachineWare GmbH
Jan Henrik Weinstock, MachineWare GmbH
Rainer Leupers, RWTH Aachen University
Motivation

- VPs indispensable in modern software development
  - Everising SW and HW complexity
- Performance often not sufficient
- Limiting factors
  - Fast models (especially CPU)
  - Modern simulation infrastructure
  - Efficient standardization
What is SIM-V?

- Fast, functional RISC-V ISS
  - Executes unmodified RISC-V binary on simulation host
  - Deep introspection and instrumentation
  - Gather simulation results
- Enables “shift left”
  - Earlier software development & test
- Design goals
  - High performance
  - Easy to use and integrate

![Diagram](image.png)
SIM-V vs. QEMU: Dhrystone

![Graph comparing SIM-V vs. QEMU Dhrystone performance for different numbers of cores.](image)
FTL - Fast Translator Library
Processor Modeling and High-Speed Dynamic Binary Translation Toolkit

• Just-in-time Binary Translation (JIT)
  • Decode target instruction
  • Translate to host instructions
  • Cache and reuse translations

• System Level Software
  • Privilege Levels
  • MMU & TLB
  • Interrupts, Traps

• Target Software Analysis
  • Debugger integration
  • Scripting interface through VCML
  • Non-intrusive register inspection
SIM-V Custom Extensions

- SIM-V’s modular architecture enables custom extensions
  - Instructions, registers, …
- SDK for extension development
  - Extension automatically loaded by SIM-V as shared library
- Leverage FTL for performance
FTL Integrations

Pure SystemC TLM-2.0

Debug PyVP CI Trace & Analysis

MachineWare Parallel VCML^4 TLM-2.0

Ethernet I^2C SPI GPIO CAN

Parallel SystemC TLM-2.0

FTL CPU

Parallel OCX^1

QEMU^2 OR1KISS^3

Virtualizer
VCML - Virtual Components Modeling Library

Rapid Virtual Platform construction and universal tool integration

• TLM Protocols
  • SPI, I²C, Ethernet, CAN, …

• Modeling Primitives
  • Ports, Registers, Properties, …

• Component Models
  • Memories, Buses, UARTs, …

• Tool Integrations
  • Debuggers, Scripting, GUI, …

Accellera SystemC

Virtual Platform

3rd Party Models
Rich-I/O Peripherals
High-Speed Processor Models

Dev-Tools
• GUI
• GDB
• ViPER
• Python
• Profiler
SIM-V VPs - High-Speed Full-System Simulation

- SystemC/TLM-based Virtual Platform
  - Timed simulation, instruction-accurate
  - Multiprocessor/multicore design
  - Executes unmodified target software

- System-level Software Development
  - Kernel, Firmware, Drivers
  - Hardware-dependant Software

- SIM-V VPs
  - Edge – rv32 with embedded I/O
  - Compute – rv64 with rich HPC I/O
  - Custom – built for you
MachineWare Virtual Platform ExploRer (ViPER)
ViPER Trace and Analysis
PyVP

- Python scripting for CI and test
- Connect to VP over network session interface
- Control, configure, inspect VP
- Easy integration with Jenkins, Gitlab CI, …

```python
import pyvp

with pyvp.connect(localhost, 1234) as vp:
    symbols = vp.cpu.get_symbols()
    bp = symbols.search("start_kernel")
    vp.cpu.set_breakpoint(bp)
    vp.run(60)
    if (vp.state() == pyvp.BREAKPOINT_HIT and
            vp.cpu.get_pc() == bp.address):
        print("TEST PASS")
    else:
        print("TEST FAIL")
```
SIM-V Use Cases

• Early software development and test
  • Utilize in VP or stand-alone to target bare-metal, hypervisor, OS, or user-space

• Reference model for comparison
  • Execute in lock-step and compare states

• Target software analysis
  • Code coverage, hot spot analysis, …

• Continuous Integration
  • Automate tests with SIM-V in Jenkins, Gitlab, …
  • Scale easily on-premise or in the cloud
Demo
SIM-V is an ultra-fast RISC-V Simulator (2x faster than Qemu)

MachineWare FTL JIT engine for customizable processor modeling

MachineWare VCML SystemC-TLM 2.0 modeling infrastructure

E-Mail: contact@mwa.re

www.machineware.de