A Comprehensive Verification Platform for RISC-V based Processors

Emre Karabulut, Design and Verification Engineer, Yonga Technology Microelectronics R&D, Istanbul, Turkey (emre.karabulut@yongatek.com)

Berk Kisinbay, Design and Verification Engineer, Yonga Technology Microelectronics R&D, Istanbul, Turkey (berk.kisinbay@yongatek.com)

Abdullah Yildiz, Design and Verification Engineer, Yonga Technology Microelectronics R&D, Istanbul, Turkey (abdullah.yildiz@yongatek.com)

Rifat Demircioglu, Design and Verification Engineer, Yonga Technology Microelectronics R&D, Istanbul, Turkey (rifat.demircioglu@yongatek.com)

accellera SYSTEMS INITIATIVE

YONGATEK

2020 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

Agenda

- Introduction
- Motivation
- Proposed Methodology
- Preliminary Results
- Conclusion
- Future Work

Introduction

- RISC-V ecosystem[1] enables SoC designers to benefit from
  - more than 100 core/SoC design and implementations
  - instruction set simulators
  - toolchains

- RISC-V ISA specification allows to extend the instruction set with specialized extensions[2,3]

- It is estimated that there will be about 62 billion RISC-V based CPU cores in the market by 2025[4,5].

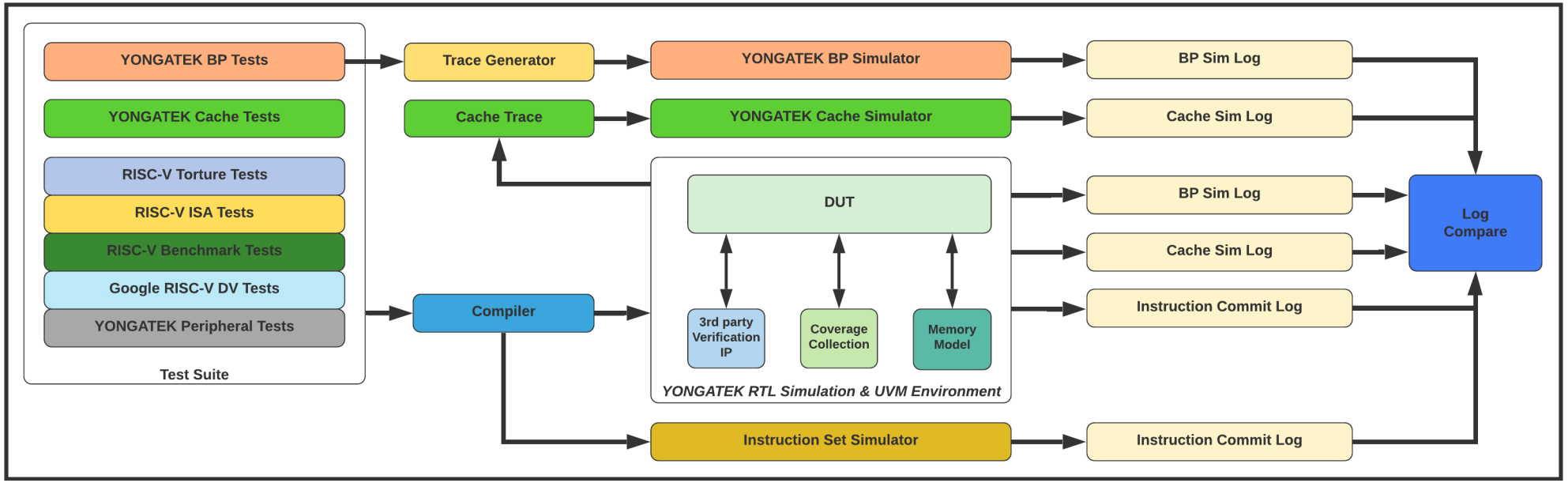
Motivation

- RISC-V's licensing model allows to reduce non-recurring engineering (NRE) costs
- There is no condition or restriction on how a RISC-V based system will be designed and implemented
- Verification of a RISC-V based SoC should not only include the verification of the core itself, but rather the verification of the whole system!

Proposed Methodology

- We present a comprehensive and UVM based RISC-V verification platform
- Our platform provides verification at both SoC, core, and block levels
- Module level verification of branch predictor and cache blocks
- Presents functional as well as stress tests in addition to standard test suites
- Provides a logging mechanism that includes committed instructions, cache transactions, branch predictor operations, etc.
- Supports debugging via OpenOCD + GDB + JTAG

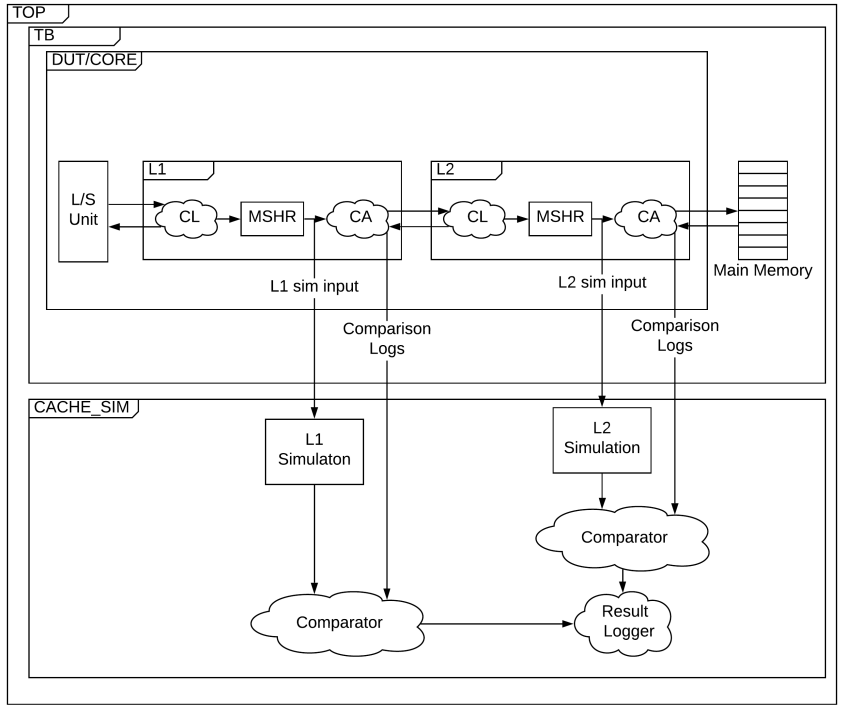
Proposed Methodology (cont'd)



YONGATEK RISC-V Verification Flow

Proposed Methodology (cont'd)

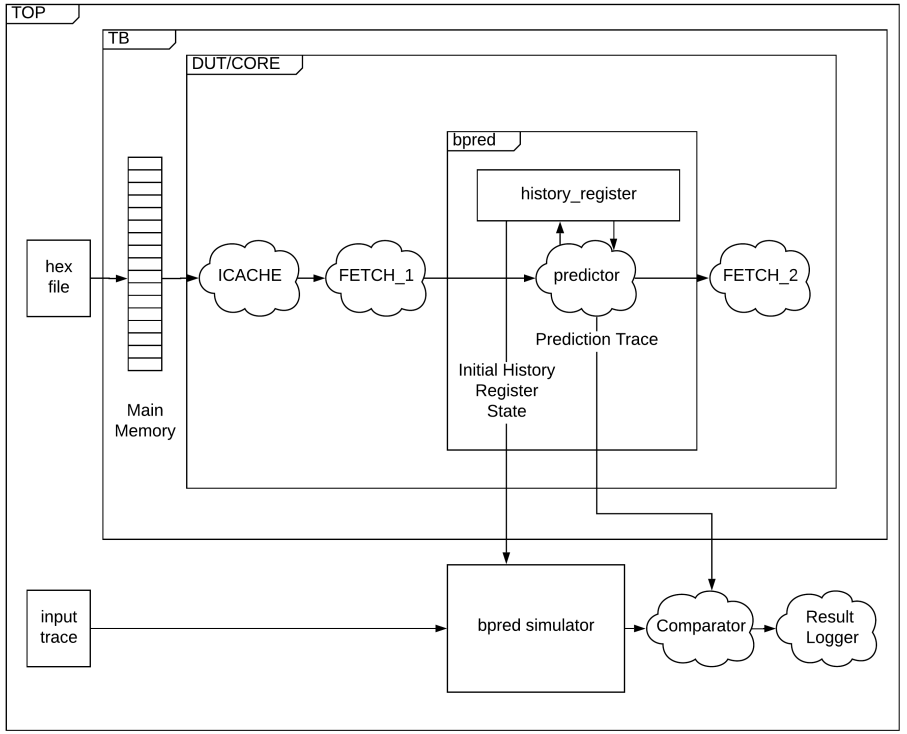
- Cache simulator
  - supports different cache configurations
  - placement: direct-mapped, set-associative, fully associative
  - replacement: random, LRU, FIFO
  - request handling: blocking/non-blocking
  - error handling: ECC
- Written in C++



Cache verification flow

Proposed Methodology (cont'd)

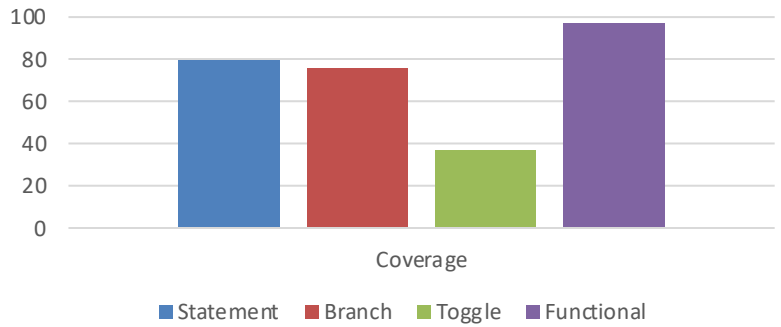
- Branch predictor simulator
  - supports bimodal, g-share, hybrid predictor models
- Written in C++



Branch predictor verification flow

Preliminary Results

- Regression consists of 300+ test cases
- Created about 10,000 covergroup elements, containing coverpoints and crosses
- Obtained on a customized SoC based on BOOM [24]



Code coverage and functional coverage

	Register Fields			
Instruction Type	Field #1	Field #2	Field #3	Field #4
LOAD	100.0	94.20	N/A	N/A
STORE	96.88	99.22	N/A	N/A
BRANCH	100.0	100.0	N/A	N/A
JALR	56.25	62.50	N/A	N/A
JAL	96.88	N/A	N/A	N/A
LOAD-FP	57.29	70.83	N/A	N/A
STORE-FP	46.88	68.75	N/A	N/A
OP	100.0	100.0	100.0	N/A
OP-32	100.0	100.0	100.0	N/A
OP-IMM	100.0	100.0	N/A	N/A
OP-IMM-32	100.0	100.0	N/A	N/A
OP-FP	94.06	92.50	90.56	N/A
AMO	12.07	4.38	13.07	N/A
(N)MADD/(N) MSUB	85.68	86.2	24.5	89.1

Instruction coverage

Conclusion

- Our verification platform
  - can support all the base instruction set and standard extensions of RISC-V ISA
  - provides verification at SoC, core, and block levels
  - gathers different features of different verification environments under a single platform

Future Work

- Trace interface and encoder support
- JTAG VIP
- Flow management GUI

Questions

References

- [1] RISC-V International, https://riscv.org/ (Accessed: 1 February 2020).
- [2] E. Alkim, H. Evkan, N. Lahr, R. Niederhagen, and R. Petri, "ISA extensions for finite field arithmetic," IACR Transactions on Cryptographic Hardware and Embedded Systems, 2020, 219-242.
- [3] J.L. Hennessy, and D.A. Patterson, "A new golden age for computer architecture," Communications of the ACM, 2019, 62.2: 48-60.
- [4] Semico Research (2019), “RISC-V market analysis: The new kid on the block,”
- https://semico.com/sites/default/files/TOC_CC315-19.pdf (Accessed: 9 May 2020).
- [5] RISC-V Foundation News (2019), “Semico forecasts strong growth for RISC-V,” https://riscv.org/2019/11/9679/ (Accessed: 9 May 2020).
- [6] T. Anderson, “Verification challenges for RISC-V adoption,” https://www.gsaglobal.org/forums/verification-challenges-for-risc-v-adoption/ (Accessed: 9 May 2020).
- [7] RISC-V Foundation, “RISC-V tests,” https://github.com/riscv/riscv-tests (Accessed: 28 July 2020).
- [8] RISC-V Compliance Task Group, “RISC-V compliance tests,” https://github.com/riscv/riscv-compliance (Accessed: 28 July 2020).
- [9] Y. Lee and H. Cook, “RISC-V torture tests,” https://github.com/ucb-bar/riscv-torture (Accessed: 28 July 2020).
- [10] RISC-V Foundation, “Spike RISC-V ISA simulator,” https://github.com/riscv/riscv-isa-sim (Accessed: 28 July 2020).
- [11] Google, “SV/UVM based instruction generator for RISC-V processor verification,” https://github.com/google/riscv-dv (Accessed: 28 July 2020).
- [12] P.D. Schiavone et al., “An open-source verification framework for open-source cores: A RISC-V case study,” International Conference on Very Large Scale Integration (VLSI-SoC), IEEE, 2018, pp. 43–48.

References (cont'd)

- [1] C. Duran et al., "Simulation and formal: The best of both domains for instruction set verification of RISC-V based processors," International Symposium on Circuits and Systems (ISCAS), IEEE, 2020, pp. 1-4.
- [14] V. Herdt, D. Große, P. Pieper, and R. Drechsler, "RISC-V based virtual prototype: An extensible and configurable platform for the system-level," Journal of Systems Architecture, 2020, 101756.
- [15] A. Chandra and M. Bartley, "A Hierarchical and Configurable Strategy to Verify RISC-V based SoCs," T&VS, 2020.
- [16] R. Molina-Robles et al., "A compact functional verification flow for a RISC-V 32I based core," Conference on PhD Research in Microelectronics and Electronics in Latin America (PRIME-LA), IEEE, 2020, pp. 1-4.
- [17] A. Oleksiak, S. Cieślak, K. Marcinek, and W.A. Pleskacz, "Design and verification environment for RISC-V processor cores," Mixed Design of Integrated Circuits and Systems, IEEE, 2019, pp. 206-209.
- [18] Symbiotic EDA, "RISC-V formal verification framework," https://github.com/SymbioticEDA/riscv-formal (Accessed: 1 July 2020).
- [19] M. Chupilko, A. Kamkin, and A. Protsenko, "Open-source validation suite for RISC-V", International Workshop on Microprocessor/SoC Test, Security and Verification (MTV), IEEE, 2019, pp. 7–12.
- [20] J. Choi, M. Vijayaraghavan, B. Sherman, A. Chlipala, Arvind, "Kami: A platform for high-level parametric hardware specification and its modular verification," Proceedings of the ACM on Programming Languages, ACM, 2017, pp. 1-30.
- [21] P. Kocher et al., "Spectre attacks: exploiting speculative execution," Symposium on Security and Privacy (SP), IEEE, 2019, pp. 1-19.
- [22] L. Moore, R. Ho, D. Letcher, K. McDermott, "RISC-V compliance & verification techniques processor cores and custom extensions", DVCon Europe, 2019.
- [23] Standard Performance Evaluation Corporation, "SPEC CPU95 Benchmarks," https://www.spec.org/cpu95 (Accessed: 10 April 2020).
- [24] K. Asanovic, D. A. Patterson, and C. Celio, "The Berkeley out-of-order machine (BOOM): an industry-competitive, synthesizable, parameterized RISC-V processor," University of California at Berkeley, 2015.