International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

Evaluation of Coverage Metrics for Assessing Test Suite Effectiveness in RISC-V Core Verification

Harinagarjun Chippagi, Dr. V. Sumalatha

Abstract— RISC-V is a popular open-source Instruction Set Architecture (ISA) that is gaining widespread adoption in the industry. The verification of a RISC-V core involves a rigorous testing process to ensure that it meets the functional requirements of the ISA. To validate the test suite of a RISC-V core verification, the following coverage metrics can be used:

1. Instruction Coverage: This metric measures the percentage of instructions in the RISC-V ISA that are exercised during the verification process. It ensures that all instructions are tested and validated, and there are no instruction-level bugs in the design.

2. Functional Coverage: This metric captures the functional requirements of the RISC-V ISA and ensures that all functionality of the core is validated. It is defined in terms of a set of properties that must be tested during the verification process.

3. Code Coverage: This metric measures the percentage of code that is executed during the verification process. It includes both the instructions that are executed and the code paths that are covered.

4. Assertion Coverage: This metric measures the percentage of assertions in the design that are exercised during the verification process. It ensures that all assertions are tested and validated, and there are no design-level bugs in the RISC-V core.

By using these coverage metrics, the verification team can ensure that the test suite for the RISC-V core verification is comprehensive and thorough, and all functional requirements are met.

Keywords— *RISC-V, verification, test suite, coverage metrics, functional coverage, code coverage, assertion coverage, instruction coverage*

I. INTRODUCTION

The RISC-V is a free and open-source Instruction Set Architecture (ISA) that is designed to be simple, modular, and scalable. The RISC-V ISA has gained significant attention in recent years due to its open-source nature, flexibility, and ease of use. The RISC-V core verification is an essential aspect of ensuring precision of the RISC-V core design. The verification of RISC-V cores is a challenging task due to the complexity and diversity of these cores. A RISC-V core consists of several modules, including the instruction fetch unit, instruction decode unit, register file, and execution units. The verification of these modules and their interactions requires a comprehensive test suite that covers all possible scenarios that can be used to test the RISC-V core's functionality, performance, and compliance with the RISC-V ISA specifications. The test suite must include both positive and negative scenarios to confirm that the core is functioning as expected.

II. SIGNIFICANCE OF COVERAGE METRICS FOR RISC-V CORE

A. Instruction Coverage

Instruction coverage measures the extent to which the test suite exercises different instructions of the RISC-V core. Instruction coverage is measured as a percentage of the instructions in the ISA that were executed during testing. It ensures that all instructions are tested and validated, and there are no instruction-level bugs in the design.

B. Functional Coverage

Functional coverage measures the percentage of functional scenarios that were tested during the RISC-V core verification process. This coverage metric confirms that the RISC-V core is functioning correctly and can handle all possible scenarios. It is defined as a set of functional coverage points that need to be covered by the test suite. These coverage points are defined based on the RISC-V ISA specification and the design of the RISC-V core.

C. Code Coverage

Code coverage is a metric that helps in understanding how much of the RISC-V core design is tested. It is a useful metric that helps to assess the quality of the RISC-V test suite. Code coverage tools will use one or more criteria to determine how the code was exercised during the execution of the test suite. The common metrics that are mentioned in coverage reports include:

- Block Coverage: A code block is a sequence of one or more statements. Block coverage provides information about how well a test suite exercises the individual blocks of code in the RISC-V core, helping to identify areas that are not tested and may contain bugs.
- Branch Coverage: It yields more precise coverage details than block coverage by obtaining coverage results for various branches in the RISC-V core individually. With branch coverage, a piece of design

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

code is considered 100% covered when each branch of a conditional statement is executed at least once.

- Statement Coverage: It describes the number of executed statements in the core.
- Expression Coverage: It measures coverage statistics for logical expressions.
- Toggle Coverage: It describes design activity in terms of changes in signal values.
- Finite State Machine Coverage: It measures the number of FSM states that are exercised and the number of state transitions that occurred during the RISC-V core verification.

D. Assertion Coverage

Assertion coverage is a software testing metric that measures the effectiveness of the assertions used in a program. Assertions are statements that are added to a program to check if certain conditions hold true at a particular point in the program's execution. It is used to determine the number of these assertions that were executed during testing. The main goal of assertion coverage is to ensure that all assertions in the code are executed at least once during the testing process. This is important because assertions help identify bugs and other issues in the code, and their execution during testing confirms that these issues are detected and fixed before the code is released. Assertion coverage is measured as a percentage of the total number of assertions in the code that were executed during testing. This metric can help software testers and developers identify areas of the code that require additional testing and find assertions that are not executed and may need to be updated or removed.

III. IMPLEMENTATION AND RESULTS

A. Verification Plan for Coverage Metrics

To define the verification plan for the RISC-V processor core, set goals that were clear, specific, measurable, and aligned with the overall project goals. Identified the coverage metrics that will be used to measure the progress of verification. Selected the coverage metrics based on the goals of verification and relevant to the design being verified. Metrics identified for a RISC-V processor core are Instruction coverage, Functional coverage, Code Coverage, Assertion coverage. After identifying the coverage metrics, define the coverage plan to specify the target coverage levels for each metric, which will be achieved through a combination of directed tests, random tests, and corner-case tests. Implemented the defined coverage plan, which involves writing test cases, creating a test bench, and running simulations. Tracked progress of verification using the coverage metrics defined in the coverage plan. After completing the simulations, the coverage results were analyzed to check whether the verification goals were met. If the coverage levels are below the target levels, then additional tests are created and run to increase coverage. In the end, coverage metrics and targets are adjusted based on the results of the analysis, and new verification methods are added to achieve the desired coverage levels.

1) Instruction Coverage

To measure instruction coverage, a test suite comprising a set of programs that exercise all instructions in the RISC-V ISA, was developed. These programs have the instruction name that is executed, which will be given as input to a function. Later, the respective input will be compared with the string array which has instruction names for that specific register mode and a cover group was defined to cover these bins.

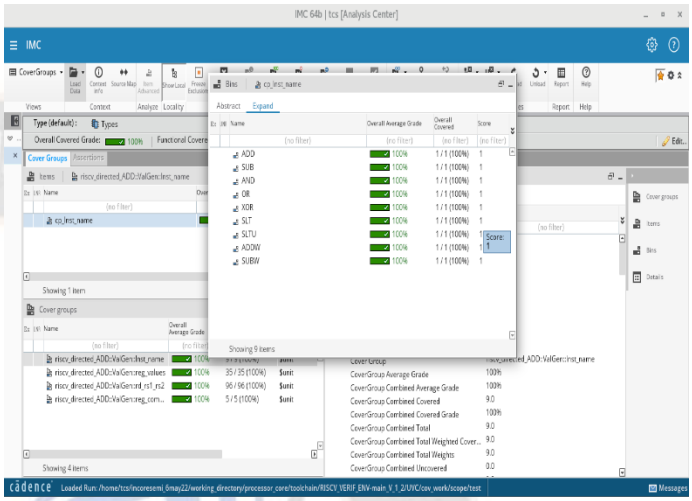

Fig. 1. Coverage for Register-Register Type Instructions

2) Functional Coverage

To analyse the functional coverage metrics for RISC-V core, the functional coverage plan was prepared in such a way that it covers all the nook and corner cases for each instruction depending on the instruction format,considering the following cover groups for validating the coverage:

- Cover group for operand selection from the General-Purpose Registers (GPRs): This checks whether the operands were assigned with every possible general-purpose register that is available in the RISC-V core. This was achieved by assigning GPRs to the operands in cyclic increments so that no two operands will be assigned with the same GPR.
- Cover group for various operand combinations: This checks whether the test suite exercised all possible register combinations. As there are three operands, all possible permutations and combinations around those operands were considered.
- Cover group for all possible operand values: This checks whether the operand was assigned with all possible values like extreme positive, positive, zero, negative, extreme negative. This was achieved by randomizing the positive and negative values in the given range from 0 to most positive value and most negative value to –1 respectively.

The above scenarios were mentioned in different cover groups to simplify understanding in the final coverage report. The instructions were also segregated depending on their register modes, to help in reducing the complexity of the code.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

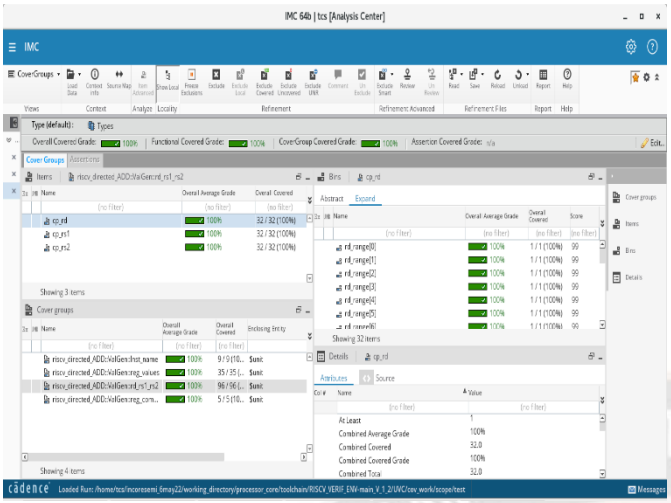

Fig. 2: Functional Coverage for Register-Register Type Instructions

3) Code Coverage

To obtain coverage analysis, Cadence xcelium tool specific commands were used to generate the coverage report, which is available in Integrated Metrics Centre (IMC). This coverage report provides the code coverage, block coverage, branch coverage, statement coverage, expression coverage, toggle coverage and FSM coverage. A coverage file which represents coverage configuration, was created for the design with .ccf extension, where the condition can be set or enable coverage through commands that are by default in disabled condition. To improve the coverage up to 100%, analyze the coverage post simulation and identify the lines which were not executed. Include the test vectors in test bench to trigger/execute those missing statements. This is how the test bench is improved and code coverage closure is achieved.

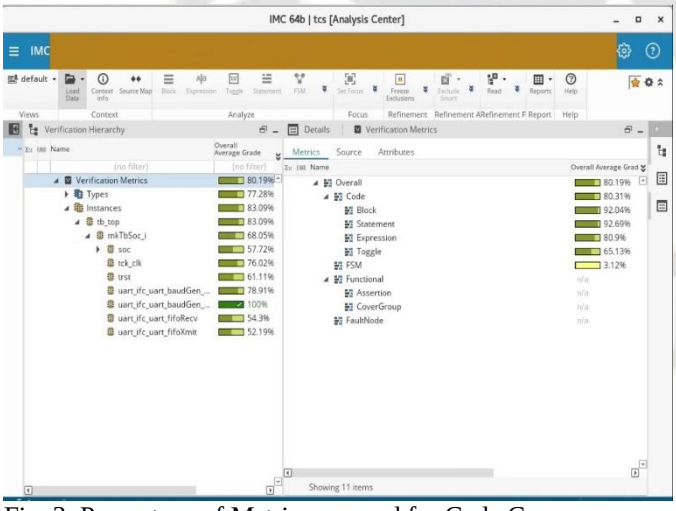

Fig. 3: Percentage of Metrics covered for Code Coverage

4) Assertion Coverage

Ddeveloped an assertion plan which is RISC-V complaint. As per the RISC-V specification manuals, the plan was categorized into two parts, one for 32 GPRs and the other for Control and Status Registers (CSRs). For GPRs, the instructions with similar opcodes are categorized. Once the instruction's opcode is known, then the code will automatically pick register values for source and destination registers according to the cases written. Once this is done, check function 3 and function 7 values, to know the distinct instruction name. Once the code runs, it goes through all cases and picks the instruction, checks for the assert statement which is written based on instruction's behavior. Check if the destination register has been updated with the desired value after operation on the source registers. Later, according to the instructions, a property must be checked for all instructions. This includes whether the PC value has been changed and instruction signal has the opcode value of respective instruction. As per the instruction signal, at the positive edge of clock, the destination register must reflect the values properly and display PASS/FAIL statement. For CSRs, most of them have a constant value which must remain same throughout the simulation and the assert statements will fail if the signal fails to be stable.

Register Check for 32 GPRs

PES	Instruction	Instruction Behaviour	Input Registers Involved	Register Polled as Output	Property to be checked	Description	PASS/FAIL
TYPE 1	ADD	ADD rd, rs1, rs2 ADD (rs1,rs2) -> rd	clk, IR, PC, rs1 and rs2	rd	1. IR and PC value is getting changed at posedge of clk 2. As per IR, rd is reflecting the proper value based on rs1 and rs2 value at the posedge of the clk.	The values in source registers are added and must be placed in destination register	If it matches the description assertion must PASS or else FAIL
	SUB	SUB rd, rs1, rs2 SUB (rs1,rs2) -> rd	clk, IR, pc, rs1 and rs2	rd	1. IR and PC value is getting changed at posedge of clk 2. As per IR, rd is reflecting the proper value based on rs1 and rs2 value at the posedge of the clk.	The content of second source register is subtracted from the first source register and must be placed in destination register	If it matches the description assertion must PASS or else FAIL
	AND	AND rd, rs1, rs2 AND (rs1,rs2) -> rd	clk, IR, pc, rs1 and rs2	rd	1. IR and PC value is getting changed at posedge of clk 2. As per IR, rd is reflecting the proper value based on rs1 and rs2 value at the posedge of the clk.	The contents of source registers are logically ANDed and result must be shown in destination register	If it matches the description assertion must PASS or else FAIL
					1. IR and PC value is getting		

Fig.4. Categorization of Assertion Plan

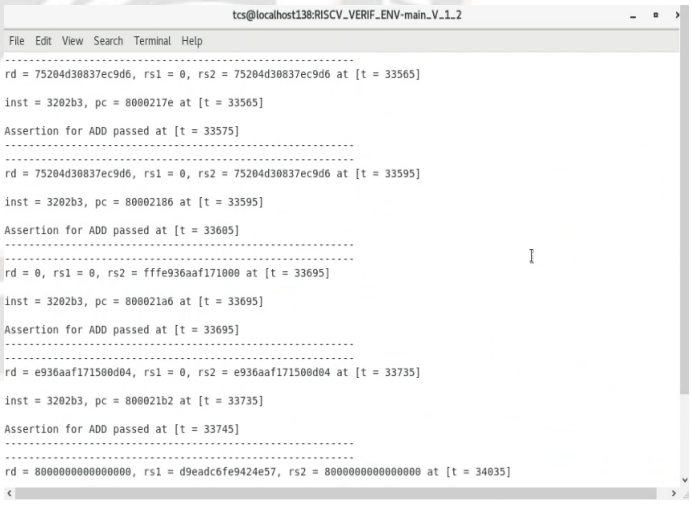

Fig. 5. Terminal Output showing Assertion Pass/Fail criteria

IV. CONCLUSION

The verification of RISC-V cores is a critical process that ensures the correctness and functionality of these cores. The test suite used for verification must be comprehensive and cover all possible scenarios that the core can encounter. Coverage metrics are used to evaluate the completeness of the test suite and determine areas that need improvement. The

coverage metrics discussed in this paper, including instruction coverage, functional coverage, code coverage, and assertion coverage, can be used to validate the test suite of a RISC-V core verification. By using these coverage metrics, the verification process can be improved, and the quality of RISC-V cores can be ensured.

REFERENCES

[1] *A. Waterman and K. Asanovic, "The RISC-V Instruction Set Manual; Volume I: Unprivileged ISA",* SiFive Inc. and CS Division, EECS Department, University of California, Berkeley, 2019.

[2] Integrated Metrics Centre User Guide 23.03

[3] Integrated Coverage User Guide 23.03 A. Waterman and K. Asanovic, "The RISC-V Instruction Set Manual; Volume I: Unprivileged ISA", SiFive Inc. and CS Division, EECS Department, University of California, Berkeley, 2019.

[4] **A. Waterman and K. Asanovic**, "The RISC-V Instruction Set Manual; Volume II: Privileged Architecture", SiFive Inc. and CS Division, EECS Department, University of California, Berkeley, 2021.

[5] **J. Hennessy and D. Patterson**, "Computer Architecture: A Quantitative Approach", 6th Edition, Morgan Kaufmann, 2017.

[6] **S. Sutherland, D. Mills, and C. Spear**, "SystemVerilog for Verification: A Guide to Learning the Testbench Language Features", 3rd Edition, Springer, 2012.

[7] **M. Keating**, "The Simple Art of SoC Design: Closing the Gap between RTL and ESL", Springer, 2011.

[8] **C. Spear and G. Tumbush**, "SystemVerilog for Verification: A Guide to Learning the Testbench Language Features", 2nd Edition, Springer, 2008.

[9] **A. Mehta**, "Comprehensive Functional Verification: The Complete Industry Cycle", Morgan Kaufmann, 2008.

[10] **J. Bergeron**, "Writing Testbenches: Functional Verification of HDL Models", 2nd Edition, Springer, 2003.

[11] **S. Palnitkar**, "Verilog HDL: A Guide to Digital Design and Synthesis", 2nd Edition, Prentice Hall, 2003.

[12] **IEEE Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language**, IEEE Std 1800-2017, 2017.

Harinagarjun Chippagi is a Senior Design Verification Lead with over 15 years of experience in VLSI design verification. He is currently pursuing his Ph.D. in Digital IC Design & Verification Methodologies at Jawaharlal Nehru Technological University, Anantapur, focusing on re-usable UVM-based complex SoC verification. His expertise spans System Verilog, UVM, and FPGA/ASIC verification, with significant experience in emulating complex SoC designs using Mentor Veloce platforms. Mr. Harinagarjun Chippagi has led various high-profile projects, including IP & SoC verification complex micro Controllers and Network on Chip (NoC) verification systems. He holds an M.Tech in Digital Systems & Computer Electronics and has earned multiple certifications in functional verification methodologies. His research interests include hardware-software co-verification, rapid prototyping, and advanced verification methodologies for complex semiconductor designs.

E-mail :arjunpartha99@gmail.com , harinagarjun.chippagi@ieee.org

Dr. V. Sumalatha is a distinguished Professor of Electronics and Communication Engineering (ECE) and the Director of Industrial Relations & Placements at Jawaharlal Nehru Technological University Anantapur (JNTUA), Andhra Pradesh, India. With a Ph.D. in Wireless Networks from JNTU Anantapur , she has over two decades of academic and administrative experience. She has held various leadership roles, including Head of the ECE Department, Coordinator for Academic & Planning, and Training and Placement Officer. Dr. Sumalatha has also served as the University Nodal Officer for MHRD's All India Survey of Higher Education (AISHE) and as the Program Coordinator for the JNTUA-Texas Instruments University Program. Her expertise spans wireless networks, digital systems, and computer electronics, and she has significantly contributed to enhancing industrial relations and student placements at JNTUA. A dedicated academician, she continues to play a pivotal role in shaping the educational and professional landscape of the institution.

E-mail : sumaatp@yahoo.com , vsumalatha.ece@jntua.ac.in