

Portland State University

ECE 593 – Pre-Silicon Validation

MultiProcessor System

Group - 9

Team Members:

Frezewd Debebe

Janvier Mpifizi Rutihunza

Sal Esmaeil

Milestone - 4 Report

UVM Architecture

1 Verification Strategy and Objectives

DUT: 4-core ALU/Memory system with shared bus arbitration.

Verification Goal: Ensure all ALU operations, memory accesses, and arbitration scenarios across all cores are functionally correct and compliant with the spec.

Result: 100% functional coverage and 0 Error.

2 Testbench Architecture

Agents: Four independent agents (core_agn1_0 to 3), each containing a sequencer, driver, and monitor.

Virtual Sequencer: Used mp_dynamic_vseq to coordinate traffic across all four cores simultaneously, simulating real-world bus contention. It managed a shared "pool" of transactions. Using a semaphore (bucket lock), cores competed for transactions, effectively stressing the DUT's internal arbiter.

Scoreboard: A centralized component with a Reference Model that calculates the expected result for every ALU and Memory operation.

Coverage: Tracks the progress of the verification plan using SystemVerilog Covergroups.

3 Test Cases Implemented

mp_test (Directed Test)

- **Focus:** Verifying basic bus handshaking and connectivity.
- **Scale:** 90 transactions.
- **Result:** 90 matches. This confirmed that the basic req/gnt/rvalid protocol was functional across all four cores.

mp_alu_test (Stress Coverage Closure)

- **Focus:** Reaching 100% functional coverage and testing the Arbiter under heavy load.
- **Scale:** 1000 transactions.
- **Result:** 946 matches, The "missing" 54 matches were filtered NOPs (Write-NOPs), which the scoreboard is intentionally designed to ignore. This test successfully filled holes in the Special and Shift opcode bins for specific cores that were missed in shorter runs.

4 Functional Coverage Analysis

CP_CORE: (100%) All 4 cores successfully gained bus access.

CP_OPCODE: (100%) Every instruction type (ALU, Logical, Shift, Memory, Special) was executed.

CP_ADDR: (100%) Memory accesses spanned the entire 2KB range (Low, Mid, and High regions).

CROSS_CORE_OP: Proved that every core executed every possible instruction type, ensuring the arbiter does not starve specific opcodes on specific cores.

5 Result (Waveform)

In the multiprocessor system, the communication protocol follows a structured handshake where a core initiates an instruction by asserting its unique request signal (`req[core_id]`), signaling to the arbiter that it requires bus access. In the waveform, the system's operation is validated at the positive clock edge where the arbiter asserts the grant signal (`gnt[core_id]`), signifying that the core has officially won bus ownership. At this precise moment of alignment, the monitor captures the stable "intent" signals—including the opcode, address, and operands (A and B) while the DUT begins processing the operation. The transaction cycle only concludes when the DUT pulses the response valid signal (`rvalid`), indicating that the resulting data is ready to be sampled from the bus.

Table 1 and 2 shows the request and grant signals and how those signals can be interpreted from the waveform.

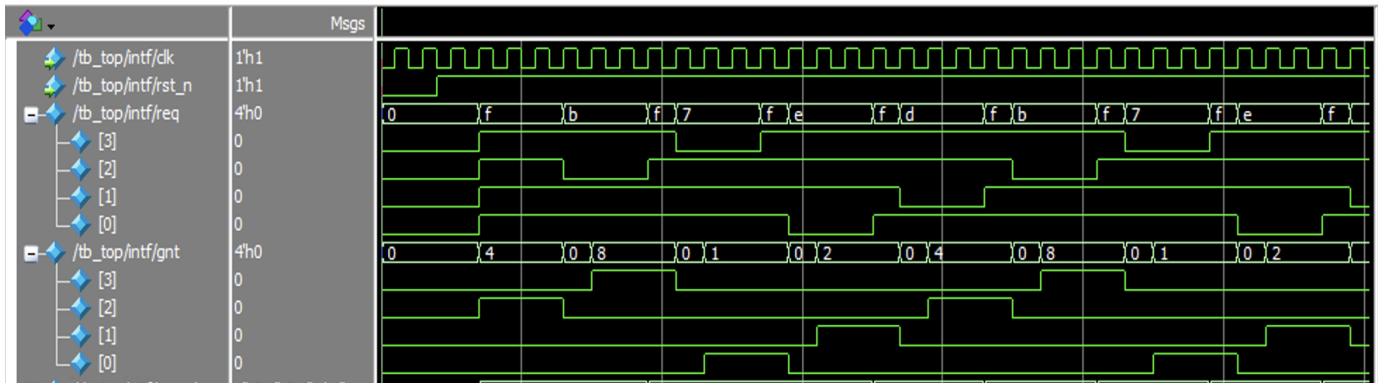


Figure 1: waveform showing req and gnt signals

6 Challenges and Solutions

Statistical Distribution: Initial runs of 200 transactions left 4-5 bins empty. Because of the 4-core arbitration, a higher volume of 1000 transactions was required to overcome statistical "unluckiness" and guarantee 100% coverage of the cross-bins.

7 Conclusion

The verification of the Multiprocessor System is complete. The combination of UVM's constrained-random capabilities and a robust scoreboard reference model has proven that the design is free of functional errors. The merged coverage reports (`func_cov_report_TOTAL`) confirm that the verification plan has been fully executed.

Table 1: Grant Signals for the 4 cores

Decimal	Binary ($C_3C_2C_1C_0$)	Core Grant
8	1000	Core 3 is active
4	0100	Core 2 is active
2	0010	Core 1 is active
1	0001	Core 0 is active

Table 2: Request Signals for the 4 cores

Hex	Binary ($C_3C_2C_1C_0$)	Core requesting
1	0001	Only Core 0
2	0010	Only Core 1
3	0011	Core 0 and 1
4	0100	Only Core 2
5	0101	Core 0 and 2
6	0110	Core 1 and 2
7	0111	Core 0, 1 and 2
8	1000	Only Core 3
9	1001	Core 0 and 3
a	1010	Core 1 and 3
b	1011	Core 0, 1 and 3
c	1100	Core 2 and 3
d	1101	Core 0, 2 and 3
e	1110	Core 1, 2 and 3
f	1111	All 4 Cores