# ECE-593: Fundamentals of Pre-Silicon Validation
Maseeh College of Engineering and Computer Science
Winter, 2025

Project Name: Multiprocessor System

Members:Janvier Mpfizi Rutihunza, Frezewd Debebe,Sal Esmaeil

Date: 01/30/2026

| | |
|---|---|
| Project Name | Multiprocessor System |
| Location | Portland State University |
| Start Date | 01/21/2026 |
| Estimated Finish Date | |
| Completed Date | |



| Prepared by: Team 9 | |
|---|---|
| Prepared for: Prof. Venkatesh Patil | |
| Team Member Name | Email |
| Janvier Mpfizi Rutihunza | jgasaba2@pdx.edu |
| Frezewd Debebe | frezewd@pdx.edu |
| Sal Esmaeil | esmaeil@pdx.edu |

| Design Features: |
| --- |
| Three-Processor Shared-Memory Architecture:<br>The system has three independent CPUs that operate in parallel and share memory space. Each processor can issue memory requests at the same time, enabling parallel execution |
| RTL Design Implementation: |

For Milestone-1 of the Multiprocessor System, the processor functionality is implemented using an RTL **Generator** approach. Instead of designing full instruction-level CPU cores, each processor is modeled as a generator that produces deterministic memory transactions. This approach enables early pre-silicon validation of the system architecture while keeping the implementation complexity manageable.

Each generator represents a processor that interacts with the shared bus, centralized arbiter, and memory subsystem. The generator issues read and write requests, allowing verification of arbitration fairness, shared bus correctness, and memory data integrity. This methodology aligns with pre-silicon validation best practices by focusing on system-level correctness before introducing full processor logic.

**Generator Functional Behavior**

After reset, each generator executes a fixed sequence:

1. Requests access to the shared bus.
2. Writes a unique data value to a unique memory address.
3. Issues a read request to the same address.
4. Captures the returned data from memory.
5. Compare the read data with the expected value.
6. Asserts a **done** signal and reports a **pass/fail** status.

This deterministic behavior ensures repeatable simulations and simplifies debugging and verification.

**Role in System Verification**

The generator-based design validates the following system components:

- Centralized round-robin arbiter behavior under simultaneous processor requests.
- Correct bus grant logic and isolation between processors.
- Proper memory read and write operations.
- End-to-end data flow between processors and shared memory.

The generator modules provide a scalable foundation that can be replaced with real processor cores in later milestones without modifying the shared bus, arbiter, or memory subsystem.

Memory Organization:
The subsystem memory is 2 KB, organized as 1 byte wide x 2K deep. All memory transactions must pass through the shared cache, ensuring consistent access patterns.

- Total memory size: 2 KB
- Width: 1 byte (8 bits)
- Depth: 2K entries
- $1KB = 1024\, bytes$
- Total bytes:
    - $2 \times 1024 = 2048\, bytes$

Since $1KB = 1024\, bytes$, the total memory size is:
$2 \times 1024 = 2048\, bytes$
Addressing

- Memory addresses range from:
    - $MEM[0{:}\, 2047]$

Single Shared Cache:
Sits between the CPUs and main memory. All Processors access the same cache structure, ensuring only one copy of each memory block exists.

- Cache Line Choice

- Cache line size: 16 bytes
- Number of Cache Lines:
  - $(2048\ bytes\ \div\ 16\ bytes\ per\ line)\ =\ 128\ \ cache\ lines$
- Main memory: 2048 individual bytes (or 2048 entries).
- Cache:
  - 128 lines, each line holding 16 bytes.
- Write policy:
  - Write-back

Centralized Arbiter for Bus Access:
Centralized Arbiter manages access to shared cache and memory bus and since all the CPUs may request access at the same time, the arbiter ensures fair and deterministic access and support for round-robin scheduling.
It performs the following functions:
• Monitors request signals from all CPUs
• Grants bus access to one CPU at a time
• Ensures fairness through a chosen policy round-robin scheduling.

Shared Bus Interconnect:
All CPUs communicate with the shared cache and memory through a single shared bus.

| Project Description: |
| --- |
| We are going to design and verify a multi-processor system that can effectively execute multiple Instructions. The hardware design consists of three processors and one memory subsystem. The design architecture is implemented using a shared-memory architecture. In the shared-memory architecture, all three processors access a single pool of memory, and communication between the processors is facilitated through shared memory access. The multi-processor system will consist of three processing units connected through a shared bus to a Memory |

Subsystem. All processing units have a common cache memory to reduce the need for data transfer between the processors and main memory.

| Important Signals/Flags |
| --- |
| |

| Design Signals |
| --- |
| |

| Block Diagram |
| --- |

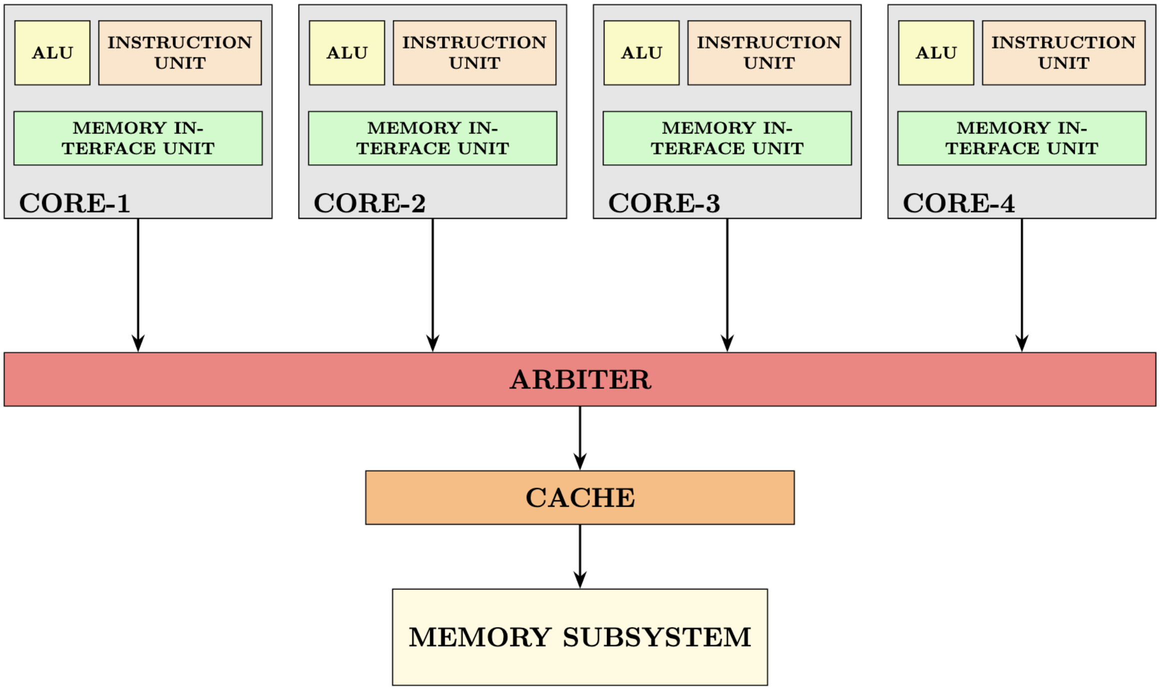


Figure 1: 4-Core Processor Architecture

| References/Citations |
| --- |
| |